SIEMENS

SIMOTION

SIMOTION SCOUT Kommunikation

Systemhandbuch

11/2010

Rechtliche Hinweise

Warnhinweiskonzept

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

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

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

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

VORSICHT
ohne Warndreieck bedeutet, dass Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG
bedeutet, dass ein unerwünschtes Ergebnis oder Zustand eintreten kann, wenn der entsprechende Hinweis nicht beachtet wird.

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

Qualifiziertes Personal

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

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

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

Marken

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

Haftungsausschluss

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

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
DEUTSCHLAND

Vorwort

Vorwort

Inhalt

Das vorliegende Dokument ist Bestandteil des **Dokumentationspaketes System- und Funktionsbeschreibungen**.

Gültigkeitsbereich

Dieses Handbuch ist gültig für SIMOTION SCOUT Produktstufe V4.2:

- SIMOTION SCOUT V4.2 (Engineering System der Produktfamilie SIMOTION),

Informationsblöcke des Handbuches

Das vorliegende Handbuch beschreibt die Kommunikationsmöglichkeiten von SIMOTION-Systemen.

- **Übersicht Kommunikationsfunktionen und -dienste**

  Allgemeine Informationen welche Kommunikationsmöglichkeiten SIMOTION hat.

- **PROFIBUS**

  Informationen zur DPV1 Kommunikation und zur Einrichtung und Programmierung der Kommunikation zwischen SIMOTION- und SIMATIC-Geräten.

- **PROFINET IO**

  Informationen zum Projektieren von PROFINET mit SIMOTION

- **Ethernet Allgemein (TCP/IP- und UDP-Verbindungen)**

  Informationen zur Einrichtung und Programmierung der Ethernet-Kommunikation zwischen SIMOTION- und SIMATIC-Geräten.

- **Routing - Kommunikation über Netzwerkgrenzen**

  Allgemeine Informationen zu Routing

- **SIMOTION IT**

  Allgemeine Informationen welchen IT- und Web-Funktionen SIMOTION zur Verfügung stellt.

- **PROFIsafe**

  Allgemeine Informationen zum Projektieren von fehlersicheren Steuerungen.

- **PROFIdrive**

  Beschreibung des PROFIdrive Profils.

- **Index**

  Stichwortverzeichnis zum Finden der Informationen

SIMOTION Dokumentation

Einen Überblick zur SIMOTION Dokumentation erhalten Sie in einem separaten Literaturverzeichnis.

Diese Dokumentation ist als elektronische Dokumentation im Lieferumfang von SIMOTION SCOUT enthalten und besteht aus 10 Dokumentationspaketen.

Zur SIMOTION Produktstufe V4.2 stehen folgende Dokumentationspakete zur Verfügung:

- SIMOTION Engineering System Handhabung
- SIMOTION System- und Funktionsbeschreibungen
- SIMOTION Service und Diagnose
- SIMOTION IT
- SIMOTION Programmieren
- SIMOTION Programmieren - Referenzen
- SIMOTION C
- SIMOTION P
- SIMOTION D
- SIMOTION Ergänzende Dokumentation

Hotline und Internetadressen

Weiterführende Informationen

Unter folgendem Link finden Sie Informationen zu den Themen:

- Dokumentation bestellen / Druckschriftenübersicht
- Weiterführende Links für den Download von Dokumenten
- Dokumentation online nutzen (Handbücher/Informationen finden und durchsuchen)

http://www.siemens.com/motioncontrol/docu

Bei Fragen zur technischen Dokumentation (z. B. Anregungen, Korrekturen) senden Sie bitte eine E-Mail an folgende Adresse:
docu.motioncontrol@siemens.com

My Documentation Manager

Unter folgendem Link finden Sie Informationen, wie Sie Dokumentation auf Basis der Siemens Inhalte individuell zusammenstellen und für die eigene Maschinendokumentation anpassen:

http://www.siemens.com/mdm

Training

Unter folgendem Link finden Sie Informationen zu SITRAIN - dem Training von Siemens für Produkte, Systeme und Lösungen der Automatisierungstechnik:

http://www.siemens.com/sitrain

FAQs

Frequently Asked Questions finden Sie in den Service&Support-Seiten unter **Produkt Support**:

http://support.automation.siemens.com

Technical Support

Landesspezifische Telefonnummern für technische Beratung finden Sie im Internet unter **Kontakt**:

http://www.siemens.com/automation/service&support

Inhaltsverzeichnis

Einleitung 1

1.1 Thema Kommunikation in der SIMOTION Dokumentation

Übersicht

Zum Thema Kommunikation finden Sie Hinweise in den einzelnen Gerätehandbüchern, in den Programmierhandbüchern und in diesem Handbuch Kommunikation.

Handbuch Kommunikation

Das Handbuch Kommunikation behandelt insbesondere Informationen, die für eine Kommunikation der SIMOTION Geräte mit Geräten außerhalb der SIMOTION Familie wichtig sind, insbesondere zur SIMATIC.

In diesem Handbuch finden Sie deshalb die Erläuterung der notwendigen Projektierungsschritte, die auf beiden Seiten der Kommunikationspartner durchgeführt werden müssen, um eine fehlerfrei funktionierende Kommunikationsbeziehung zu erhalten.

Deshalb wird in diesem Handbuch auch sehr intensiv auf die Einstellungen und die Programmierung der SIMATIC S7 Stationen als Kommunikationspartner der SIMOTION eingegangen.

Gerätehandbücher und Programmierhandbücher

Die Gerätehandbücher behandeln das Thema Kommunikation insbesondere aus der Sicht der Geräte selbst, d. h. bzgl. der elektrischen Eigenschaften der vorhandenen Schnittstellen sowie deren Einstellmöglichkeiten mit dem Engineering-System SIMOTION SCOUT.

Weitere Informationen finden Sie auch in den Handbüchern **Modulare Maschinenkonzepte** und **Basisfunktionen**, die Teil des SIMOTION Dokumentationspaketes sind.

Hier finden Sie keine Hinweise, wie Ihre Partnerstationen einzustellen sind.

Übersicht Kommunikations-Funktionen und -Dienste 2

2.1 Netzwerkmöglichkeiten

2.1.1 Einleitung

Als integraler Bestandteil der "Totally Integrated Automation" (TIA) bieten die SIMOTION und SIMATIC Netzwerklösungen die für die Kommunikationsanforderungen Ihrer Anwendung erforderliche Flexibilität und Leistungscharakteristik, ganz gleich wie einfach oder komplex Ihre Anwendung sein mag.

Hinweis

Dieser Abschnitt. beschreibt allgemein, welche Kommunikations-Funktionen und Dienste es innerhalb der Siemens Automatisierungstechnik gibt. Das bedeutet nicht zwangsläufig, dass auch alle genannten Funktionen für SIMOTION zur Verfügung stehen. Die Details zu den von SIMOTION unterstützten Funktionen finden Sie in den Kapiteln 4 - 8.

SIMOTION und SIMATIC Netzwerke für jede Anwendung

Die SIMOTION Produkte unterstützen eine Vielzahl von Netzwerkmöglichkeiten. Mit diesen Netzwerklösungen können Sie die SIMOTION Geräte entsprechend den Anforderungen Ihrer Anwendung kombinieren.

Zur weiteren Optimierung der Netzwerklösungen bieten SIMOTION Produkte integrierte Kommunikationsdienste bzw. -funktionen für die Erweiterung der Leistungsfähigkeit des Netzwerkprotokolls.

2.1.2 PROFINET

Übersicht

PROFINET basiert auf dem offenen Standard Industrial Ethernet für die Industrieautomatisierung zur unternehmensweiten Kommunikation und erweitert die Fähigkeit zum Datenaustausch Ihrer Automatisierungskomponenten bis in die Office-Umgebung, sodass Sie Ihre Automatisierungskomponenten, sogar die dezentralen Feldgeräte und Antriebe, an Ihr Local Area Network (LAN) anschließen können.

Weil PROFINET alle Ebenen in Ihrer Organisation miteinander verbindet – von den Feldgeräten bis zu den Managementsystemen – können Sie das anlagenweite Engineering mittels herkömmlichen IT-Standards umsetzen. Wie bei jeder auf Industrial Ethernet basierenden Lösung unterstützt PROFINET elektrische, optische und drahtlose Netzwerke.

Weil PROFINET auf Industrial Ethernet basiert und es sich nicht um eine abgewandelte "PROFIBUS for Ethernet"-Implementierung handelt, kann PROFINET die vorhandene installierte Basis von Ethernet-kompatiblen Geräten ausnutzen. Auch wenn PROFINET kein Master/Slave-System ist, bieten die Kommunikationsdienste PROFINET IO und PROFINET CBA die von Automatisierungssystemen geforderte Funktionalität:

- Mit PROFINET IO können Sie dezentrale Feldgeräte (z. B. digitale oder analoge Signalbaugruppen) und Antriebe direkt an ein Industrial Ethernet-Subnetz anschließen.
- PROFINET CBA (Component Based Automation) unterstützt modulare Lösungen für die Maschinen- und Anlagenkonstruktion. Sie definieren Ihr Automatisierungssystem in Form von autonomen Komponenten, wobei jede Komponente aus unabhängigen und in sich geschlossenen Aufgabenstellungen besteht.

Beide Kommunikationsdienste bieten Echtzeitfunktionalität, um PROFINET zu einer Echtzeitimplementierung zu machen. Außerdem erlaubt PROFINET das gleichzeitige Vorhandensein der Echtzeitkommunikation Ihres Automatisierungsprozesses und Ihrer sonstigen IT-Kommunikation, zur gleichen Zeit im selben Netzwerk, ohne dass dadurch das Echtzeitverhalten Ihres Automatisierungssystems beeinträchtigt wird.

Zur weiteren Unterstützung von fehlersicheren bzw. "sicherheitsrelevanten" Anwendungen kommuniziert das PROFIsafe-Profil mit den fehlersicheren Geräten über das PROFINET-Subnetz.

2.1.3 Industrial Ethernet

Übersicht

Dadurch, dass Industrial Ethernet ein Kommunikationsnetzwerk für die Verbindung von Befehlsebene und Zellenebene bereitstellt, können Sie mit Industrial Ethernet die Fähigkeiten für den Datenaustausch Ihrer Automatisierungskomponenten in die Office-Umgebung ausdehnen.

Industrial Ethernet basiert auf den Normen IEEE 802.3 und IEEE 802.3u für die Kommunikation zwischen Computern und Automatisierungssystemen und ermöglicht es Ihrem System dadurch, große Datenmengen über lange Entfernungen auszutauschen.

2.1.4 PROFIBUS

Übersicht

PROFIBUS basiert auf den Normen IEC 61158 / EN 50170 und bietet eine Lösung mit offenem Feldbus für die komplette Produktions- und Prozessautomatisierung. PROFIBUS bietet schnellen und zuverlässigen Datenaustausch und integrierte Diagnosefähigkeiten. PROFIBUS unterstützt herstellerunabhängige Lösungen mit dem weltweit größten Fremdhersteller-Support. Für Ihr PROFIBUS-Subnetz können Sie eine Vielzahl von Übertragungsmedien nutzen: elektrisch, optisch und drahtlos.

PROFIBUS umfasst die folgenden Kommunikationsdienste:

- PROFIBUS DP (Decentralized Peripherals) ist ein Kommunikationsprotokoll, das sich besonders gut für die Produktionsautomatisierung eignet.
  PROFIBUS DP bietet schnellen, zyklischen und deterministischen Austausch von Prozessdaten zwischen einem Bus-DP-Master und den zugeordneten DP-Slavegeräten. PROFIBUS DP unterstützt die isochrone Kommunikation. Die synchronisierten Ausführungszyklen stellen sicher, dass die Daten in konsistenten äquidistanten Zeitabständen übertragen werden.
- PROFIBUS PA (Process Automation) erweitert PROFIBUS DP und bietet eigensichere Daten- und Leistungsübertragung entsprechend der Norm IEC 61158-2.
- PROFIBUS FMS (Fieldbus Message Specification) ist für die Kommunikation auf Zellebene ausgelegt, wo die Steuerungen miteinander kommunizieren. Mittels PROFIBUS FMS können Automatisierungssysteme verschiedener Hersteller miteinander kommunizieren.
- PROFIBUS FDL (Fieldbus Data Link) wurde für die Übertragung von mittleren Datenmengen optimiert, um die fehlerfreie Übertragung von Daten auf dem PROFIBUS-Subnetz zu unterstützen.

Außerdem nutzt PROFIBUS Profile, um Kommunikationsmöglichkeiten für die Anforderungen bestimmter Anwendungen zu bieten, z. B. PROFIdrive (für die Bewegungssteuerung) oder PROFIsafe (für fehlersichere bzw. "sicherheitsrelevante" Anwendungen).

2.1.5 MPI (mehrpunktfähige Schnittstelle)

Übersicht

MPI sind integrierte Schnittstellen für SIMOTION und SIMATIC Produkte (SIMOTION Geräte, SIMATIC S7 Geräte, SIMATIC HMI sowie SIMATIC PC und PG).

MPI bietet eine Schnittstelle zur PG/OP-Kommunikation. MPI bietet auch einfache Netzwerkfähigkeit unter Verwendung der folgenden Dienste: Kommunikation über globale Daten (GD), S7-Kommunikation und S7-Basiskommunikation.

Das elektrische Übertragungsmedium für MPI nutzt die Norm RS 485, die auch von PROFIBUS verwendet wird.

2.1.6 Punkt-zu-Punkt-Kommunikation (PtP)

Übersicht

SIMOTION Geräte können programmiert werden, damit sie Daten mit einer anderen Steuerung im Netzwerk austauschen. Auch wenn die Punkt-zu-Punkt-Kommunikation nicht als Subnetz eingestuft ist, bietet die Punkt-zu-Punkt-Verbindung die serielle Übertragung, z. B. über RS232 oder RS485, von Daten zwischen zwei Stationen, z. B. mit einer SIMATIC-Steuerung oder sogar mit einem Fremdgerät, das kommunikationsfähig ist.

Für die Punkt-zu-Punkt-Kommunikation können Sie CP-Baugruppen (z. B. eine CP340) oder ET200-Baugruppen verwenden, um Daten zwischen zwei Steuerungen zu lesen und zu schreiben. Die Punkt-zu-Punkt-Kommunikation stellt damit eine leistungsstarke und kostengünstige Alternative zu Buslösungen dar, insbesondere dann, wenn nur wenige Geräte an das SIMOTION-Gerät angeschlossen werden sollen.

Die Punkt-zu-Punkt-Kommunikation bietet die folgenden Fähigkeiten:

- Anpassung an das Protokoll des Kommunikationspartners mittels Standardverfahren oder ladbaren Treibern.
- Definieren eines benutzerspezifischen Verfahrens mittels ASCII-Zeichen
- Kommunikation mit anderen Arten von Geräten wie Bedienstationen, Druckern oder Kartenlesegeräten

Weitere Literatur

Weiterführende Literatur zur Punkt-zu-Punkt-Kommunikation finden Sie in den Beschreibungen der CP- bzw. ET200-Baugruppen.

2.2 Kommunikationsdienste (oder Netzwerkfunktionen)

2.2.1 Einleitung

SIMOTION und SIMATIC Geräte unterstützen einen Satz spezifischer Kommunikationsdienste, die die Datenpakete steuern, die über die physikalischen Netzwerke übertragen werden. Jeder Kommunikationsdienst definiert einen Satz Funktionen und Leistungsmerkmale, z. B. die zu übertragenden Daten, die zu steuernden Geräte, die zu beobachtenden Geräte und die zu ladenden Programme.

Kommunikationsdienste der SIMOTION und SIMATIC Produkte

Kommunikationsdienste, die auch häufig als Netzwerkfunktionen bezeichnet werden, sind die Softwarekomponenten, die die physikalische Hardware der Netzwerke nutzen. Softwareschnittstellen (z. B. S7-Systemfunktionen) im Endgerät (z. B. SIMOTION Gerät, SIMATIC S7 Gerät oder PC) bieten Zugriff auf die Kommunikationsdienste. Eine Softwareschnittstelle verfügt jedoch nicht unbedingt über alle Kommunikationsfunktionen für den Kommunikationsdienst. Ein solcher Dienst kann im jeweiligen Endsystem mit unterschiedlichen Softwareschnittstellen zur Verfügung gestellt werden.

2.2.2 PG/OP-Kommunikationsdienste

Übersicht

PG/OP-Dienste sind die integrierten Kommunikationsfunktionen, mit denen SIMATIC und SIMOTION Automatisierungssysteme mit einem Programmiergerät (z. B. STEP 7) und Bedien- und Beobachtungsgeräten kommunizieren. Alle SIMOTION und SIMATIC Netzwerke unterstützen die PG/OP-Kommunikationsdienste.

2.2.3 S7-Kommunikationsdienste

Übersicht

S7-Kommunikationsdienste bieten Datenaustausch mittels Kommunikations-Systemfunktionsbausteinen (SFBs) und Kommunikations-Funktionsbausteinen (FBs) für konfigurierte S7-Verbindungen.

Alle SIMOTION Geräte und SIMATIC S7 Geräte haben integrierte S7-Kommunikationsdienste, mit denen das Anwenderprogramm in der Steuerung das Lesen oder Schreiben von Daten auslösen kann. Diese Funktionen sind von spezifischen Netzwerken unabhängig, sodass Sie die S7-Kommunikation über jedes Netzwerk (MPI, PROFIBUS, PROFINET oder Industrial Ethernet) programmieren können.

Für die Übertragung von Daten zwischen den Steuerungen müssen Sie eine Verbindung zwischen den beiden Steuerungen konfigurieren. Die integrierten Kommunikationsfunktionen werden vom SFB/FB in der Anwendung aufgerufen. Sie können bis zu 64 KB Daten zwischen SIMOTION und SIMATIC S7 Geräten übertragen.

Sie können mit Ihrem HMI-Gerät, Programmiergerät (PG) oder Computer auf die Daten in der Steuerung zugreifen, weil die S7-Kommunikationsdienste in das Betriebssystem der SIMOTION Geräte und SIMATIC S7 Gerät integriert sind. Diese Art von Peer-to-Peer-Verbindung benötigt keine zusätzliche Verbindungseinrichtung. (Wenn Sie jedoch eine Verbindung zu einem dieser Geräte konfigurieren, können Sie über die symbolischen Namen auf die Daten zugreifen.)

Hinweis

SFBs können bei SIMOTION nicht verwendet werden.

2.2.4 S7-Basiskommunikationsdienste

Übersicht

S7-Basiskommunikationsdienste bieten Datenaustausch mittels Kommunikations-Systemfunktionen (SFCs) für nicht konfigurierte S7-Verbindungen. Diese SFCs (z. B. X_GET oder X_PUT) lesen oder schreiben die Daten auf ein SIMATIC Steuerung, um so kleine Datenmengen über ein MPI-Subnetz an eine andere S7-Station (S7-Steuerung, HMI oder PC) zu übertragen.

Die SFCs für die S7-Basiskommunikation kommunizieren nicht mit Stationen in anderen Subnetzen. Für die S7-Basiskommunikation brauchen Sie keine Verbindungen zu konfigurieren. Die Verbindungen werden aufgebaut, wenn das Anwenderprogramm die SFC aufruft.

Hinweis

Die S7-Basiskommunikationsdienste können Sie nur über eine MPI-Verbindung zwischen SIMATIC S7-300, S7-400 oder C7-600 Steuerungen nutzen.

2.2.5 Kommunikationsdienst "Globale Daten"

Übersicht

Neben den anderen Möglichkeiten für die Netzwerkkommunikation können Sie eine Kommunikationsverbindung 'Globale Daten' (GD) konfigurieren, um die zyklische Datenübertragung zwischen SIMATIC Steuerungen bereitzustellen, die an ein MPI-Netzwerk angeschlossen sind. Der Datenaustausch läuft im Rahmen des normalen Prozessabbildaustauschs ab, weil die globale Datenkommunikation in das Betriebssystem der SIMATIC-Steuerung integriert ist.

Der Empfang der globalen Daten wird nicht quittiert, weil es sich bei der globalen Datenkommunikation um ein Verfahren zum Übertragen von Daten handelt. Ein Publisher (Datenquelle) sendet die Daten an einen oder mehrere Subscriber (Datensenke), und die Subscriber empfangen die Daten. Der Publisher empfängt keine Quittierung von den Subscribern darüber, dass diese die übertragenen Daten erhalten haben.

Hinweis

Die globale Datenkommunikation können Sie nur über eine MPI-Verbindung zwischen SIMATIC S7-300, S7-400 oder C7-600 Steuerungen nutzen.

Die GD-Kommunikation erfordert keine besondere Programmierung und auch keine Programmbausteine in Ihrem STEP 7-Anwenderprogramm. Die Betriebssysteme der einzelnen Steuerungen bearbeiten den Austausch von globalen Daten. Mit STEP 7 konfigurieren Sie eine globale Datentabelle (GD) mit dem Quellpfad der Daten, die an die Subscriber übertragen werden sollen. Diese GD-Tabelle wird mit der Hardware-Konfiguration für den Publisher und die Subscriber geladen.

Globale Daten sind für SIMOTION nicht verfügbar.

2.2.6 PROFINET-Kommunikationsdienste

Übersicht

PROFINET umfasst die folgenden Kommunikationsdienste:

- Mit dem Kommunikationsdienst PROFINET IO können Sie E/A-Geräte und Antriebe über Ethernet-Physik an die SIMOTION oder SIMATIC Steuerung anbinden. Mit PROFINET IO kann das in der Steuerung ausgeführte Anwenderprogramm die Eingangs- und Ausgangsdaten der E/A-Geräte und Antriebe verarbeiten. Die Adressierung für PROFINET IO konfigurieren Sie in STEP 7 bzw. SIMOTION SCOUT.
- Mit PROFINET CBA können Sie Ihr Automatisierungssystem nach autonomen Untereinheiten oder Komponenten definieren. Bei diesen Komponenten kann es sich um PROFINET IO-, PROFIBUS DP- oder auch Fremdgeräte und -subnetze handeln.

Wenn Sie die Kommunikationsdienste PROFINET CBA für eine komponentenbasierte Lösung einsetzen möchten, konfigurieren Sie die SIMATIC-Steuerungen und die E/A-Geräte in einzelnen Komponenten in STEP 7. Anschließend konfigurieren Sie die Kommunikation zwischen den verschiedenen Komponenten mit SIMATIC iMAP.

Beide Kommunikationsdienste PROFINET IO und PROFINET CBA bieten die von Automatisierungssystemen geforderte Echtzeitkommunikation.

Hinweis

PROFINET CBA steht nur für SIMATIC-Geräte zur Verfügung, jedoch nicht für SIMOTION Geräte.

2.2.7 Industrial Ethernet-Kommunikationsdienste

Übersicht

Industrial Ethernet basiert auf den Normen IEEE 802.3 und IEEE 802.3u und verbindet die Automatisierungssysteme mit Ihrem Business-System, sodass Sie im Büro Zugang zu den Daten haben und diese bearbeiten können.

Industrial Ethernet umfasst die folgenden Kommunikationsdienste:

- Die ISO-Übertragung bietet Dienste zum Übermitteln von Daten über Verbindungen, die die fehlerfreie Datenübertragung unterstützen. ISO-Übertragung ist nur mit STEP7 möglich.
- Mit TCP/IP können Sie zwischen Steuerungen und Computern in PROFINET oder Industrial Ethernet-Netzwerken zusammenhängende Datenblöcke austauschen. Bei TCP/IP sendet die Steuerung zusammenhängende Datenblöcke.
- ISO-on-TCP (RFC 1006) unterstützt die fehlerfreie Datenübertragung. Bei SIMOTION nur, wenn über den SCOUT ONLINE gegangen wird. Falls die Kommunikation aus dem Anwenderprogramm erfolgt, muss RFC selbst programmiert werden.
- UDP (User Datagram Protocol) und UDP Multi-Cast bietet einfache Datenübertragung ohne Quittierung. Sie können zusammenhängende Datenblöcke von einer Station zu einer anderen übertragen, z. B. zwischen einer SIMOTION und SIMATIC Steuerung, einem PC oder einem Fremdsystem.
- Bei der IT-Kommunikation (Informationstechnologie) können Sie mittels Standard-Ethernet Protokollen und Diensten wie z. B. FTP, HTTP und E-Mail über PROFINET- oder Industrial Ethernet-Netzwerke Daten gemeinsam nutzen.

2.2.8 PROFIBUS-Kommunikationsdienste

Übersicht

PROFIBUS umfasst die folgenden Kommunikationsdienste:

- PROFIBUS DP (Distributed Peripherals) fördert die transparente Kommunikation mit der dezentralen Peripherie. Das SIMOTION/STEP 7-Anwenderprogramm greift genauso auf die dezentrale Peripherie zu wie auf die E/A im zentralen Baugruppenträger der Steuerung (bzw. der SPS). PROFIBUS DP ermöglicht die direkte Kommunikation mit der dezentralen Peripherie. PROFIBUS DP entspricht den Normen EN 61158 / EN 50170.
- PROFIBUS PA (Process Automation) erleichtert die direkte Kommunikation mit Instrumenten der Prozessautomatisierung (PA). Dies umfasst sowohl den zyklischen Zugriff auf die E/A, typischerweise mit einem SPS-Master, als auch den azyklischen Zugriff auf den potenziell umfangreichen Satz von Gerätebetriebsparametern, typischerweise mit einem Engineering-Werkzeug wie Process Device Manager (PDM). PROFIBUS PA entspricht der Norm IEC 61158

- PROFIBUS FMS (Fieldbus Message Specification) ermöglicht die Übertragung von strukturierten Daten (FMS-Variablen). PROFIBUS FMS entspricht der Norm IEC 61784.
- PROFIBUS FDL (Fieldbus Data Link) wurde für die Übertragung von mittleren Datenmengen optimiert, um die fehlerfreie Übertragung von Daten auf dem PROFIBUS-Subnetz zu unterstützen. PROFIBUS FDL unterstützt die SDA-Funktion (Daten mit Quittierung senden)

Hinweis

SIMOTION Geräte unterstützen ausschließlich den Kommunikationsdienst PROFIBUS DP.

Für die fehlersichere Kommunikation nutzen SIMOTION und SIMATIC Geräte das Profil PROFIsafe für PROFIBUS DP.

Für die Kommunikation zwischen SIMOTION Geräte hin zu den angeschlossenen Antrieben nutzen die SIMOTION Geräte das Profil PROFIdrive.

Weitere Literatur

Eine Gegenüberstellung der SIMATIC S7 und SIMOTION Systemfunktionen finden Sie im Verzeichnis 2_FAQ auf der Utilities & Applications CD.

2.3 Weitere Verfahren zum Austauschen von Informationen

Neben den Standardkommunikationsnetzwerken unterstützen SIMOTION und SIMATIC auch weitere Mittel zur gemeinsamen Nutzung von Informationen über Netzwerke.

Gemeinsame Datennutzung mit anderen Anwendungen über OPC (OLE for Process Control)

Mit OPC (OLE for Process Control) können Windows-Anwendungen auf Prozessdaten zugreifen, sodass auf einfache Weise Geräte und Anwendungen verschiedener Hersteller miteinander kombiniert werden können. OPC bietet nicht nur eine offene, herstellerunabhängige Schnittstelle, sondern auch eine bedienerfreundliche Client/Server-Konfiguration für den standardisierten Datenaustausch zwischen (z. B. HMI- oder Office-Anwendungen), die kein spezifisches Netzwerk oder Protokoll benötigen.

Der OPC-Server bietet Schnittstellen für den Anschluss der OPC-Client-Anwendungen. Sie konfigurieren die Client-Anwendung für den Zugriff auf Datenquellen, z. B. Adressen im Speicher einer SPS. Weil mehrere verschiedene OPC-Clients gleichzeitig auf denselben OPC-Server zugreifen können, können dieselben Datenquellen für jede OPC-konforme Anwendung genutzt werden.

Neben OPC-Servern bietet SIMATIC NET auch Anwendungen zum Konfigurieren und Testen von OPC-Verbindungen: Advanced PC Configuration (APC) und OPC Scout (dient zum Testen und in Betrieb nehmen einer OPC-Anwendung bzw. eines OPC-Servers). Mit diesen Werkzeugen verbinden Sie SIMOTION und SIMATIC S7-Produkte mit anderen OPC-konformen Anwendungen.

Die SIMATIC NET OPC-Server unterstützen die folgenden Kommunikationsdienste:

- PROFINET IO (mittels PROFINET- oder Industrial Ethernet-Subnetz)
- PROFINET CBA (mittels PROFINET- oder Industrial Ethernet-Subnetz)
- TCP/IP (mittels PROFINET- oder Industrial Ethernet-Subnetz)
- PROFIBUS DP (mittels PROFIBUS-Subnetz)
- PROFIBUS FMS (mittels PROFIBUS-Subnetz)
- S7-Kommunikation
- S5-kompatible Kommunikation

Gemeinsame Datennutzung in einer Office-Umgebung mittels Informationstechnologie (IT)

SIMOTION und SIMATIC nutzen die herkömmlichen IT-Werkzeuge (wie E-Mail - nur SIMATIC, HTTP-Webserver, FTP und SNMP) mit PROFINET und Industrial Ethernet-Netzwerken, um die Fähigkeiten zur gemeinsamen Datennutzung in der Office-Umgebung zu erweitern.

Für die SIMOTION-Geräte werden die entsprechenden Funktionen über SIMOTION IT DIAG zur Verfügung gestellt, siehe SIMOTION IT Ethernet-basierende HMI- und Diagnosefunktionen.

PROFIBUS 3

3.1 PROFIBUS Kommunikation

3.1.1 PROFIBUS Kommunikation (Übersicht)

Beschreibung

PROFIBUS DP (Decentralized Peripherals) ist für den schnellen Datenaustausch in der Feldebene konzipiert. Die Kommunikation erfolgt zwischen einem PROFIBUS-Master Klasse 1 (z. B. einem SIMOTION-Controller) und PROFIBUS-Slaves (z. B. einem SINAMICS S120 Antrieb). Der Datenaustausch mit den dezentralen Geräten erfolgt vorwiegend zyklisch (DP-V0-Kommunikation). Hierbei liest die zentrale Steuerung (SIMOTION-Controller) die Eingangsinformationen zyklisch von den Slaves und schreibt die Ausgangsinformationen zyklisch an die Slaves. Weiter werden über die zyklischen Dienste Diagnosefunktionen zur Verfügung gestellt. Die folgende Grafik zeigt das Datenprotokoll am PROFIBUS DP.

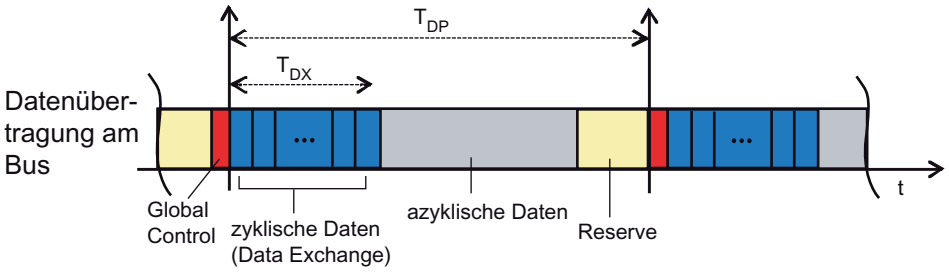


Bild 3-1 Datenprotokoll am PROFIBUS

3.2 Kommunikation mit SIMATIC S7

3.2.1 Mögliche Kommunikationsverbindungen zwischen SIMOTION und SIMATIC

Im Folgenden werden die Möglichkeiten beschrieben, wie ein SIMOTION und ein SIMATIC S7-Gerät über PROFIBUS miteinander kommunizieren können.

Dazu gibt es verschiedene Möglichkeiten:

- Ein SIMOTION-Gerät wird als DP-Slave an ein DP-Mastersystem einer SIMATIC S7 angebunden.
- Ein SIMATIC S7-Gerät wird als DP-Slave an ein DP-Mastersystem einer SIMOTION angebunden.
- Zwischen SIMOTION und SIMATIC S7 wird eine Master-Master-Kommunikation verwendet.

Bei der Ankopplung als DP-Slave unterscheidet man nochmals 2 Varianten:

- Die Anbindung als Norm-Slave mittels einer GSD-Datei.
- Die Anbindung als intelligenter DP-Slave (I-Slave).

Als I-Slave werden Stationen bezeichnet, die eine eigenständige Intelligenz besitzen, und erst durch ihre eigene Programmierung in ihrem Funktionsumfang als DP-Slave festgelegt werden.

D. h., dass diese Stationen zuerst bzgl. Ihrer Kommunikationsstruktur fertig projektiert sein müssen, bevor sie als I-Slave weiterverwendet werden können.

Die verfügbaren I-Slaves finden Sie im HW-Katalog von HW Konfig im Ordner "bereits projektierte Stationen".

Unterschied: "Normaler" DP-Slave (Norm-Slave) - Intelligenter DP-Slave (I-Slave)

Bei einem "normalen" DP-Slave, wie z. B. einem kompakten (ET 200eco) oder modularen (ET 200M) DP-Slave, greift der DP-Master auf die dezentralen Ein-/Ausgänge zu.

Bei einem intelligenten DP-Slave greift der DP-Master nicht auf die angeschlossenen Ein-/Ausgänge des Intelligenten DP-Slaves, sondern auf einen Übergabebereich im Ein-/Ausgangsadressraum der "vorverarbeitenden CPU" zu. Das Anwenderprogramm der vorverarbeitenden CPU muss für den Austausch der Daten zwischen Operandenbereich und Ein-/Ausgängen sorgen.

Hinweis

Die projektierten E-/A-Bereiche für den Datenaustausch zwischen Master und Slaves dürfen nicht von E-/A-Baugruppen "besetzt" sein.

3.2.2 SIMOTION als DP-Slave an einer SIMATIC S7

3.2.2.1 Einleitung

Im Folgenden werden die Möglichkeiten beschrieben, wie ein SIMOTION-Gerät als PROFIBUS-DP-Slave an ein PROFIBUS-Netzwerk angekoppelt werden kann.

Dazu gibt es 2 Möglichkeiten:

- Das SIMOTION-Gerät wird als Norm-Slave mittels einer GSD-Datei an das DP-Mastersystem angebunden.
- Das SIMOTION-Gerät wird als so genannter Intelligenter DP-Slave (I-Slave) in das DP-Mastersystem eingebunden

3.2.2.2 SIMOTION als DP-Slave mithilfe einer GSD-Datei an eine SIMATIC S7 koppeln

Vorgehensweise

Die GSD-Dateien für die verschiedenen SIMOTION-Plattformen müssen zunächst in STEP7 HW Konfig importiert werden.

Die entsprechenden GSD-Dateien finden Sie auf der SIMOTION SCOUT DVD "Add-on" im jeweiligen Geräteverzeichnis unter Firmware und Version.

Tabelle 3- 1 GSD-Datei

Gerät	Name der GSD Datei
SIMOTION C	Si0480aa.gsd
SIMOTION D	Si0280ab.gsd (Diese Datei kann für alle SIMOTION D verwendet werden)
SIMOTION P	Si0380fa.gsd

Nachdem diese GSD-Dateien über das Menü Extras - GSD Datei installieren in STEP7 HW Konfig importiert wurden, erscheinen die Geräte im HW-Katalog unter weitere Feldgeräte - SPS - SIMATIC - SIMOTION und können von dort in ein DP-Mastersystem einer S7-Station eingefügt werden.

Hinweis

Auf SIMOTION Geräte, die per GSD-Datei an eine SIMATIC S7 angekoppelt wurden, kann nicht mit SIMOTION SCOUT über eine geroutete Verbindung zugegriffen werden.
Der Name einer GSD-Datei ist versionsabhängig, z. B. S10180AA und S10280AA.

Hinweis

Über einen Netzwerknoten hinweg kann auch auf Antriebe, welche als Einzelantrieb eingefügt wurden, geroutet werden.

Somit kann auf SIEMENS-Antriebe, welche im SCOUT/STARTER projektiert werden können, auch geroutet werden, wenn diese als GSD-Slave / GSDML-Device in HW Konfig projektiert werden. Hier gilt die Einschränkung, dass mittels Einstellung der Online-Zugangsparameter (**Zielgerät > Onlinezugang**) ein Netzübergangpunkt anhand der Subnetz ID eingestellt werden kann.

Außerdem ist der Name der GSD-Datei versionsabhängig vergeben.

3.2.2.3 SIMOTION als I-Slave an eine SIMATIC S7 koppeln

Voraussetzung

- Auf dem Engineering-PC müssen SIMOTION SCOUT und damit STEP7 installiert sein.
- Die SIMATIC S7 und die SIMOTION Station müssen sich im gleichen Projekt befinden.

Sind diese Voraussetzungen gegeben, so kann die SIMOTION auch als so genannter I-Slave an das PROFIBUS DP Netz der SIMATIC angebunden werden.

Vorgehensweise

Es ist empfehlenswert, dass die SIMOTION Station als DP-Slave fertig projektiert ist, bevor sie als Slave am DP-Strang der SIMATIC CPU platziert wird.

Im Folgenden ist die Vorgehensweise für eine SIMOTION C beschrieben. Die Vorgehensweise ist bis auf die Auswahl der SIMOTION Plattform gleich.

1. Konfigurieren einer Station als DP-Slave z. B. SIMOTION C2xx
   Doppelklicken Sie auf die gewünschte Schnittstelle (z. B. DP2/MPI) in der Konfigurationstabelle und wählen Sie im Register Betriebsart die Option DP-Slave.
2. Konfigurieren der lokalen E/A-Adressen
   Im Register **Konfiguration** können Sie die lokalen E/A-Adressen und die Diagnoseadresse einstellen.
3. Wechseln Sie zu der projektierten SIMATIC-Station, die DP-Master für die SIMOTION sein soll.
4. Anlegen eines I-Slave
   Ziehen Sie den Stationstyp "C2xx/P3xx/D4xx-I-Slave" aus dem Fenster Hardware Katalog (Ordner bereits projektierte Stationen) per Drag & Drop auf das Symbol für das DP-Mastersystem der SIMATIC-Station.

5. Festlegen des intelligenten DP_Slave
   Doppelklicken Sie auf das Symbol für den Intelligenten DP-Slave SIMOTION und wählen das Register **Kopplung**. In diesem Register treffen Sie die Zuordnung, welche Station hier den intelligenten DP-Slave repräsentieren soll. In diesem Dialog finden Sie alle Stationen, die bereits im Projekt vorhanden sind, und als mögliche Koppelpartner in Betracht kommen.

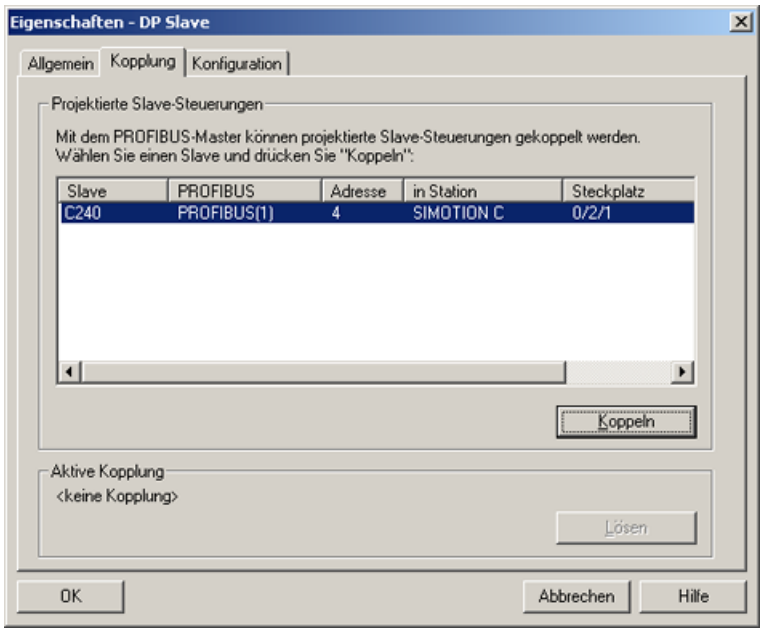


Bild 3-2 Eigenschaften DPSlave

6. Wählen Sie hier die entsprechende SIMOTION aus und drücken Sie auf **Koppeln**. Damit ist die projektierte SIMOTION Station nun als intelligenter DP-Slave an der SIMATIC angebunden

7. Wählen Sie das Register **Konfiguration** und ordnen Sie die Adressen einander zu:

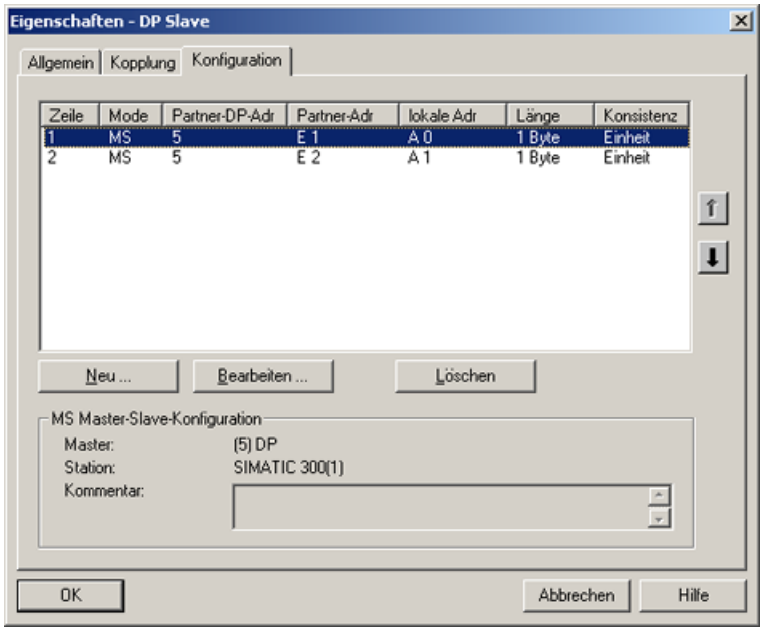


Bild 3-3 Eigenschaften - Konfiguration

   – Für den Datenaustausch mit dem DP-Master über E/A-Bereiche wählen Sie den Mode **MS** (Master-Slave)
   – Für den direkten Datenaustausch mit einem DP-Slave oder DP-Master wählen Sie den Mode **DX** (Direct Data Exchange)

8. Bestätigen Sie die Einstellungen mit **OK**.

Damit ist die Projektierung der SIMOTION Station als intelligenter DP-Slave an der SIMATIC-Station abgeschlossen und die Daten können über die angegebenen E/A-Adressen ausgetauscht werden.

3.2.3 SIMATIC S7 als DP-Slave an einer SIMOTION

3.2.3.1 Einleitung

Im Folgenden werden die Möglichkeiten beschrieben, wie eine SIMATIC-Station als PROFIBUS-DP-Slave an einem PROFIBUS-Netzwerk angekoppelt werden kann.

Dazu gibt es 2 Möglichkeiten:

- Die SIMATIC-Station wird als Normslave mittels einer GSD-Datei an das DP-Mastersystem einer SIMOTION angebunden.
- Die SIMATIC-Station wird als so genannter I-Slave in das DP-Mastersystem einer SIMOTION eingebunden.

3.2.3.2 SIMATIC als DP-Slave mithilfe einer GSD-Datei an ein SIMOTION Gerät koppeln

Vorgang

Die GSD-Dateien für die verschiedenen SIMATIC-Stationen müssen zunächst in STEP7 HW Konfig importiert werden.

Die entsprechenden GSD-Dateien finden Sie im Produktsupport unter: http://support.automation.siemens.com/ww/view/en/113652.

Nachdem diese GSD-Dateien über das Menü Extras - GSD Datei installieren in STEP7 HW Konfig importiert wurden, erscheinen die Geräte im HW-Katalog unter weitere Feldgeräte - SPS - SIMATIC und können von dort in ein DP-Mastersystem einer SIMOTION Station eingefügt werden.

Auf SIMATIC S7 Geräte, die per GSD-Datei an ein SIMOTION-Gerät angekoppelt wurden, kann nicht mit STEP7 über eine geroutete Verbindung zugegriffen werden.

3.2.3.3 SIMATIC S7 CPU als I-Slave an ein SIMOTION Gerät koppeln

Voraussetzungen

- Auf dem Engineering-PC ist SIMOTION SCOUT und damit auch SIMATIC STEP7 installiert.
- Die SIMATIC S7- und die SIMOTION-Station müssen sich im gleichen Projekt befinden

Sind diese Voraussetzungen gegeben, so kann die SIMATIC auch als so genannter I-Slave an das PROFIBUS-DP Netz der SIMOTION angebunden werden.

Vorgehensweise

Es ist empfehlenswert, dass die SIMATIC-Station als DP-Slave fertig projektiert ist, bevor sie als Slave am DP-Strang der SIMOTION platziert wird.

Im Folgenden ist die Vorgehensweise für eine CPU 315-2 DP beschrieben. Die Vorgehensweise ist bis auf die Auswahl der CPU-Typen auch bei einer S7-400 gleich.

1. Konfigurieren Sie eine Station z. B. mit der CPU 315-2 DP als DP-Slave. Doppelklicken Sie auf die Zeile 2.1 (Schnittstelle) in der Konfigurationstabelle und wählen Sie die Option DP-Slave im Register **Betriebsart**.
2. Im Register **Konfiguration** können Sie die lokalen E/A-Adressen und die Diagnoseadresse einstellen
3. Wechseln Sie zu der projektierten SIMOTION Station, die DP-Master für die SIMATIC sein soll.
4. Ziehen Sie den entsprechenden Stationstyp CPU 31x bzw. CPU 41x aus dem Fenster Hardware Katalog (Ordner bereits projektierte Stationen) per Drag & Drop auf das Symbol für das DP-Mastersystem der SIMOTION Station.

5. Doppelklicken Sie auf das Symbol für den Intelligenten DP-Slave und wählen das Register Kopplung. In diesem Register treffen Sie die Zuordnung, welche Station hier den Intelligenten DP-Slave repräsentieren soll. In diesem Dialog finden Sie alle Stationen, die bereits im Projekt vorhanden sind, und als möglicher Koppelpartner in Betracht kommen.

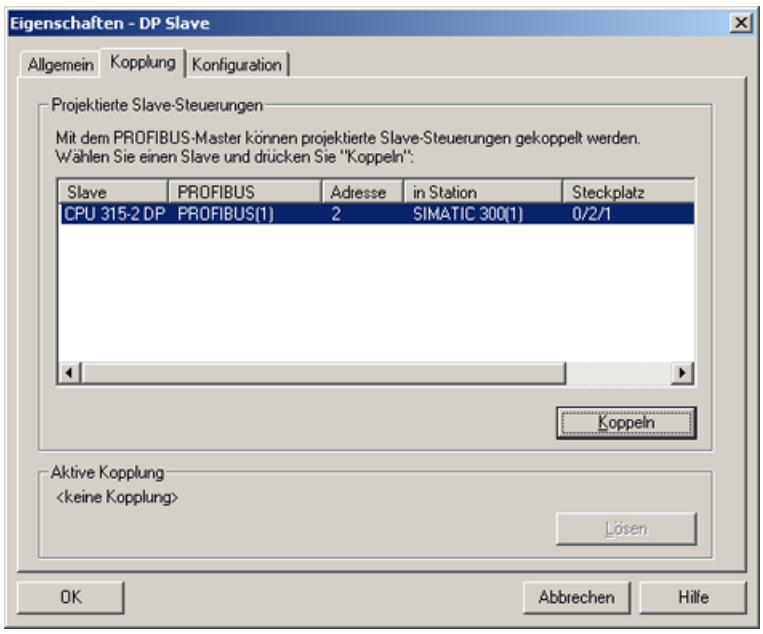


Bild 3-4 Eigenschaften - Kopplung

6. Wählen Sie hier die entsprechende S7-Station aus und drücken Sie auf **Koppeln**. Damit ist die projektierte S7-Station nun als intelligenter DP-Slave an der SIMOTION angebunden.

7. Wählen Sie das Register **Konfiguration** und ordnen Sie die Adressen einander zu:

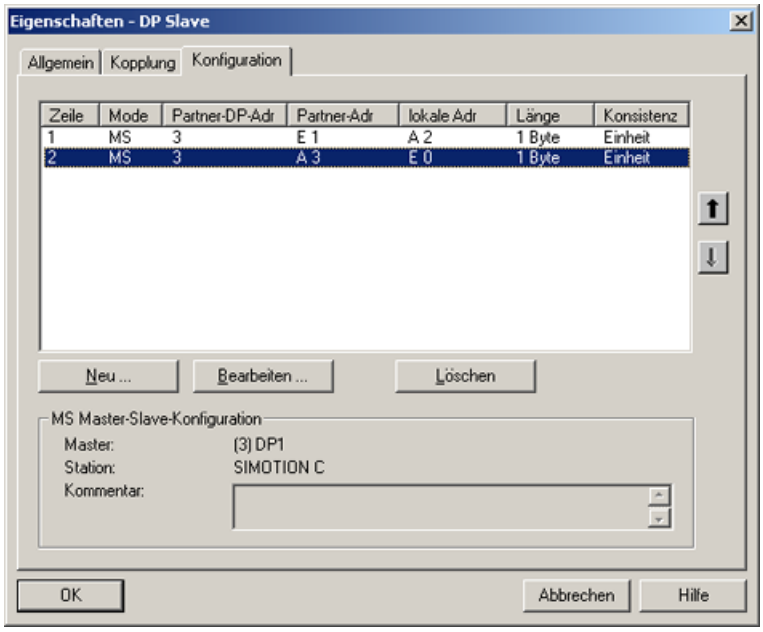


Bild 3-5 Konfiguration - Adressauswahl

   – Für den Datenaustausch mit dem DP-Master über E/A-Bereiche wählen Sie den Mode **MS** (Master-Slave)
   – Für den direkten Datenaustausch mit einem DP-Slave oder DP-Master wählen Sie den Mode **DX** (Direct Data Exchange)

8. Bestätigen Sie die Einstellungen mit **OK**.

Damit ist die Projektierung der SIMATIC-Station als intelligenter DP-Slave an der SIMOTION Station abgeschlossen und die Daten können über die angegebenen E/A-Adressen ausgetauscht werden.

3.2.4 PROFIBUS-Master-Master-Verbindung zwischen SIMATIC und SIMOTION

3.2.4.1 Einleitung

Master-Master-Kommunikation

Eine Master-Master-Kommunikations-Verbindung zwischen einem SIMATIC S7 und einem SIMOTION-Gerät über PROFIBUS wird durch den Einsatz der Systemfunktionen SFC65 (XSEND) und SFC66 (XRECEIVE) auf SIMATIC-Seite und den Systemfunktionen _Xsend und _Xreceive auf SIMOTION-Seite erstellt. Eine Projektierung der Kommunikations-Verbindung in NetPro ist nicht erforderlich.

Tabelle 3- 2 Master-Master Kommunikation

Protokoll	SIMATIC Gerät	Funktion	SIMOTION Gerät	Funktion
PROFIBUS	S7-300 CPU S7-400 CPU	SFC65 (XSEND) SFC66 (XRCV)	SIMOTION C SIMOTION D SIMOTION P	_Xsend _Xreceive

Die Vergabe der PROFIBUS-Adressen erfolgt in HW Konfig. Alle weiteren Bausteinparameter werden vom Anwender für die Verbindung festgelegt und beim Aufruf der Funktion mit übergeben. Somit verhält es sich mit der PROFIBUS-Verbindung zwischen SIMATIC und SIMOTION ähnlich wie mit einer TCP-/IP-Verbindung zwischen einer SIMATIC-Station mit integrierter Ethernetschnittstelle und einem SIMOTION-Gerät bzw. umgekehrt. Die für die Kommunikation wichtigen Parameter werden vom Anwender bestimmt und beim Baustein- bzw. Funktionsaufruf übergeben.

Im folgenden Kapitel wird auf die Parametrierung der Systemfunktionen auf SIMATIC S7-Seite und der Funktionen auf SIMOTION-Seite näher eingegangen.

3.2.4.2 SIMATIC S7-Systemfunktionen für eine PROFIBUS-Verbindung

Einleitung

Die PROFIBUS-Verbindung zwischen einer SIMATIC S7-Station und einem SIMOTION-Gerät wurde im vorhergehenden Kapitel vorgestellt. Im Folgenden soll nun die Parametrierung der SIMATIC S7-Systemfunktionen bzw. der SIMOTION-Funktionen für eine PROFIBUS-Verbindung näher erläutert werden.

SIMATIC S7-Systemfunktionen

Zur Kommunikation zwischen einer SIMATIC S7-Station und einem SIMOTION-Gerät werden auf SIMATIC S7-Seite die beiden Systemfunktionen SFC65 X_SEND und SFC66 X_RCV verwendet.
SIMOTION Funktionen:

```
CALL "X_SEND"
     REQ:=M1.0
     CONT:=FALSE            //Dies ist die DP-Adresse des
     DEST_ID:=W#16#2        //Kommunikationspartners (SIMOTION P350)
     REQ_ID:=DW#16#2        //Die REQ_ID muss mit der MessageID auf
     SD:=P#DB100.0DBX0.0 BYTE 10 //der SIMOTION-Empfang-Seite
     //übereinstimmen!
     RET_VAL:=MW64
     BUSY:=M1.1
```

Parametrierung der Systemfunktion SFC65 X_SEND

Um Daten über eine PROFIBUS-Verbindung von einer SIMATIC S7-Station an ein SIMOTION-Gerät zu versenden, wird auf SIMATIC S7-Seite die Systemfunktion SFC65 X_SEND aufgerufen.

Über den Parameter REQ wird die Datenübertragung gesteuert. d.h. wird der Parameter auf den Wert 1 gesetzt, so wird die Datenübertragung gestartet. Falls zu diesem Zeitpunkt noch keine Verbindung zum Kommunikationspartner besteht, wird sie vor dem Senden der Daten aufgebaut.

Der Parameter CONT dient der Parametrierung des Verhaltens der Verbindung nach Beendigung der Datenübertragung. Wird in den Parameter CONT der Wert 1 eingetragen, so wird die Verbindung nach Beendigung der Datenübertragung gehalten. Wird als Wert 0 eingetragen, so wird nach erfolgter Datenübertragung die Verbindung abgebaut.

Der Parameter DEST_ID enthält die PROFIBUS-Adresse des SIMOTION-Gerätes. Sie wird in STEP 7 HW Konfig festgelegt.

REQ_ID kennzeichnet die Sendedaten. d. h., über den Wert im Parameter REQ_ID können die gesendeten Daten im SIMOTION-Gerät eindeutig der S7-Station zugeordnet werden. Der hier vergebene Wert wird im Parameter messageid in der Empfangsfunktion auf SIMOTION-Seite zurück gemeldet.

Mit SD wird der Bereich festgelegt, aus dem die Sendedaten stammen.

Die beiden Parameter RET_VAL und BUSY dienen der Beobachtung des Status des Sendevorgangs. BUSY zeigt an, ob der Sendeauftrag noch läuft oder schon vollständig ausgeführt wurde. Über RET_VAL ist insbesondere im Fehlerfall eine detailliertere Diagnose möglich.

```
CALL "X_RCV"
     EN_DT:=M0.0
     RET_VAL:=MW50
     REQ_ID:=MD52
     NDA:=M0.1
```

```
RD:=P#DB110.DBX0.0 BYTE 10
```

Aufruf-Beispiel der Systemfunktion SFC66 X_RCV

Sollen auf einer SIMATIC S7-Station Daten von einem SIMOTION-Gerät empfangen werden, so ist im S7-Programm die Systemfunktion SFC66 X_RCV aufzurufen.

Am Eingang "EN_DT" der Systemfunktion wird angegeben:

- ob die Funktion nur auf das Eintreffen neuer Daten überprüfen soll (EN_DT=0) oder
- ob die empfangenen Daten nach Empfang aus der Warteschlange in den durch "RD" vorgegebenen Bereich umkopiert werden sollen (EN_DT=1).

Mit dem Parameter RET_VAL kann der Anwender den Status des Funktionsaufrufs beobachten. Speziell im Fehlerfall erhält der Anwender ausführlichere Informationen zur Ursache des Fehlers.

REQ_ID kennzeichnet die Empfangsdaten, d. h., über den Parameter REQ_ID können die empfangenen Daten einem SIMOTION-Gerät eindeutig zugeordnet werden. Der hier empfangene Wert entspricht dem Wert im Parameter messageid in der entsprechenden Sendefunktion auf SIMOTION-Seite.

Der Parameter NDA zeigt an, ob neue Daten empfangen wurden. Wenn NDA auf 1 steht, so stehen neue Daten zur Verfügung und können in den Empfangsdatenbereich übertragen werden. Hat NDA den Wert 0, so sind keine neuen Daten vorhanden.

Mit dem Parameter RD wird angegeben, wo die empfangenen Daten abgelegt werden.

SIMOTION-Funktionen

```
RetVal_PB_Senden:=
    _xsend(PB_Senden_CommunicationMode, PB_Senden_Address,
    PB_Senden_MessageID, PB_Sender_NextCommand, PB_Senden_CommandID,
    PB_Sende_Daten, PB_Sende_Daten_Laenge);
```

Beispiel für den Aufruf der SIMOTION-Funktion_xsend

Kommunizieren die SIMATIC S7-Station und das SIMOTION-Gerät über PROFIBUS, so wird auf SIMOTION-Seite zum Senden die Funktion _Xsend aufgerufen.

Hinweis

Das Senden (_Xsend()) an einen SIMOTION DP-Slave ist nur möglich, wenn der Haken **Programmieren, Status/Steuern oder andere PG-Funktionen und nichtprojektierte Kommunikationsverbindugen möglich** gesetzt ist.

Mit dem Parameter "communicationmode" wird der aufgerufenen Funktion mitgeteilt, was mit der Verbindung nach erfolgreicher Datenübertragung passieren soll. Der Datentyp der Funktion sieht die Vergabe der Werte ABORT_CONNECTION oder HOLD_CONNECTION vor. Wird der Parameter mit ABORT_CONNECTION vorbelegt, so wird die Verbindung nach der Datenübertragung abgebaut. Mit dem Wert HOLD_CONNECTION wird die Funktion so parametriert, dass nach erfolgreicher Datenübertragung die Verbindung bestehen bleibt.

Hinter dem Parameter address verbirgt sich eine Struktur vom Datentyp StructXsendDestAddr, die wiederum aus verschiedenen Parametern zusammengesetzt ist. Diese Struktur enthält alle Informationen zur Adresse des Kommunikationspartners des SIMOTION-Gerätes.

Parameter-Struktur "StructXsendDestAddr

Im Folgenden sollen die einzelnen Parameter der Struktur aufgeführt und erläutert werden.

Mit dem Parameter deviceid wird der jeweiligen SIMOTION-Hardware Rechnung getragen. Mit dem Parameter wird der physikalische Anschlusspunkt der Verbindung angegeben. So wird für eine SIMOTION C2 für die Schnittstelle X8 der Wert 1 eingetragen. Für die Schnittstelle X9 wird der Wert 2 angegeben. Ist die SIMATIC S7-Station an X101 einer SIMOTION P angeschlossen, so wird in den Parameter deviceid der Wert 1 vergeben. Für die Schnittstelle X102 wird in den Parameter deviceid der Wert 2 geschrieben. Bei der SIMOTION D wird für die Schnittstelle X126 der Wert 1 und für X136 der Wert 2 in den Parameter deviceid eingetragen.

Da für die Kommunikation über MPI oder PROFIBUS keine Subnetzmaske angegeben wird, wird der Parameter remotesubnetidlength mit dem Wert 0 vorbelegt. Als Konsequenz ergibt sich, dass die Belegung des Parameters remotesubnetid irrelevant ist.

Der Parameter remotestaddrlength wird für die MPI- bzw. PROFIBUS-Kommunikation mit dem Wert 1 belegt.

Mit dem Parameter nextstaddrlength wird die Länge der Router-Adresse angegeben. Da für die MPI- oder PROFIBUS-Kommunikation zwischen der SIMATIC S7 Station und dem SIMOTION-Gerät keine Router eingesetzt werden, wird für diesen Parameter der Wert 0 angegeben. Folglich ist auch der Parameter nextstaddr ohne Relevanz (siehe auch unten).

Der folgende Parameter remotesubnetid bezeichnet die Subnetzmaske und hat, wie bereits oben erwähnt, für die Kommunikation über MPI oder PROFIBUS keine Bedeutung.

Mit dem Parameter remotestaddr wird die eigentliche Zieladresse angegeben. Bei dem Parameter handelt es sich um ein Array. Allerdings wird für die MPI- oder PROFIBUS-Kommunikation nur der erste Index verwendet. Die weiteren fünf Indizes haben keinerlei Bedeutung.

Der Parameter nextstaddr dient der Angabe der Router-Adresse. Für ihn gilt dasselbe wie für den Parameter remotesubnetid. Auch seine Belegung hat für die Kommunikation über MPI oder PROFIBUS keine Relevanz.

Zur Identifizierung des SIMOTION-Gerätes auf der Empfangsseite wird der Parameter messageid durch den Anwender belegt. Der eingetragene Wert ermöglicht auf der SIMATIC S7-Station eine Zuordnung über den Parameter REQ_ID. Dort kann der Wert aus dem Parameter messageid ausgelesen werden.

Für diese Funktion wird mit dem Parameter nextcommand das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar oder beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

Beim Aufruf der Funktion wird ihr im Parameter commandid eine systemweit eindeutige Nummer vergeben, um den Befehlsstatus verfolgen zu können.

Die Sendedaten werden über die Variable data der Funktion beim Aufruf übergeben.

Die Länge der aus dem Sendebereich zu übertragenden Daten wird über den Parameter datalength angegeben.

Der Rückgabewert der Funktion _xsend an das Anwenderprogramm hat den Datentyp DINT und über die unterschiedlichen Rückgabewerte wird auf Probleme bei der Ausführung der Funktion hingewiesen. Ebenso wird zurückgemeldet, wenn die Daten erfolgreich gesendet wurden.

```
RetVal_PB_Empfanen:=
    _xreceive(PB_Empfangen_MessageID,
    PB_Empfangen_NextCommand,PB_Empfangen_CommandID);
```

Aufruf-Beispiel der SIMOTION-Funktion_xreceive

Das Beispiel zeigt die Verwendung der Funktion _xreceive. Die Funktion wird verwendet, wenn Daten von SIMATIC S7 Station über PROFIBUS empfangen werden sollen.

Zur Identifizierung der S7 Station, von der die Daten empfangen werden sollen, wird der Funktion _xreceive der Parameter messageid übergeben. Es wird der Wert eingetragen, der auf der S7-Seite im Parameter REQ_ID der korrespondierenden Systemfunktion _xsend vergeben wurde.

Für diese Funktion wird mit dem Parameter nextcommand das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar oder beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

Beim Funktionsaufruf wird im Parameter commandid eine systemweit eindeutige Nummer vergeben, um den Befehlsstatus verfolgen zu können.

Die Struktur, die von der Funktion ans Anwenderprogramm zurückgeliefert wird, beinhaltet die Parameter functionresult, datalength und data. Über den Parameter functionresult kann der Status des Empfangens abgefragt werden. Der Parameter datalength meldet die Anzahl der empfangenen Nutzdatenbytes nach erfolgreichem Aufruf der Funktion _xreceive zurück. Über den Parameter data kann auf die empfangenen Nutzdaten zugegriffen werden.

PROFINET IO 4

4.1 PROFINET IO Übersicht

4.1.1 PROFINET IO

Im Maschinenbau ist ein deutlicher Trend zu vermehrt dezentral konzipierten Maschinenkonzepten und mechatronischen Lösungen zu beobachten. Damit steigen die Anforderungen an die Antriebsvernetzung deutlich. Eine größere Zahl von Antrieben und kürzere Zykluszeiten, sowie die Nutzung von IT-Mechanismen, gewinnen zunehmend an Bedeutung.

Unter dem Namen PROFINET IO werden die beiden Erfolgskonzepte PROFIBUS DP und Ethernet zusammengeführt. PROFINET IO setzt auf langjähriger und erfolgreicher Erfahrungen mit PROFIBUS DP auf und verbindet gewohntes Anwenderhandling mit der gleichzeitigen Nutzung von innovativen Konzepten der Ethernet-Technologie. Die sanfte Migration von PROFIBUS DP in die PROFINET-Welt ist dabei sichergestellt.

PROFIBUS DP ist ein Bussystem bei dem zu einem Zeitpunkt nur ein Teilnehmer auf den Bus sendend zugreifen (Halbduplex-Betrieb) darf. Bei PROFINET IO wird auf die auch bei Ethernet genutzte Switching-Technologie gesetzt. Dadurch werden alle Netzwerksegmente voneinander entkoppelt und das gleichzeitige Senden und Empfangen (Vollduplex-Betrieb) auf allen Leitungen ist möglich. Damit kann das Netz durch gleichzeitige Datenübertragung mehrerer Teilnehmer wesentlich effektiver genutzt werden. Weiterhin wurde die Bandbreite auf 100 MBit/s gesteigert.

Hinweis

Detaillierte Beschreibungen zum Thema PROFINET finden Sie in dem Systemhandbuch *SIMATIC PROFINET Systembeschreibung*.

4.1.2 Applikationsmodell

Bei der Entwicklung von PROFINET IO wurde besonderer Wert auf den Investitionsschutz für Anwender und Gerätehersteller gelegt. Die Migration auf PROFINET IO erfolgt unter Beibehaltung des Applikationsmodells. Die Prozessdatensicht bleibt verglichen mit PROFIBUS DP vollständig erhalten auf:

- I/O-Daten (Zugriff auf Peripherie-Daten über logische Adressen)
- Datensätzen (Ablage von Parametern und Daten) und
- Anbindung an ein Diagnosesystem (Meldung von Diagnoseereignissen, Diagnosepuffer)

Konkret bedeutet dies, dass im Anwenderprogramm die bekannte Sicht für den Zugriff auf Prozessdaten verwendet wird. Bestehendes Programmier-Know-how kann weiterhin genutzt werden. Dies gilt auch für Geräteprofile, wie z. B. PROFIdrive, dass auch bei PROFINET IO verfügbar ist.

Auch die Engineering-Sicht bietet das gewohnte "Look and Feel". Das Engineering der dezentralen Peripherie erfolgt in gewohnter Weise mit den gleichen Tools, wie sie bei PROFIBUS bereits verwendet wurden.

4.1.3 IO-Controller

Der PROFINET IO-Controller entspricht in seiner Funktion dem Master bei PROFIBUS DP. Der IO-Controller z. B. eine SIMOTION D 4x5-2 DP/PN mit Onboard PROFINET-Schnittstelle tauscht zyklisch Daten mit dem ihm zugeordneten Peripheriegeräten (PROFINET IO-Devices) z. B. SINAMICS S120 aus.

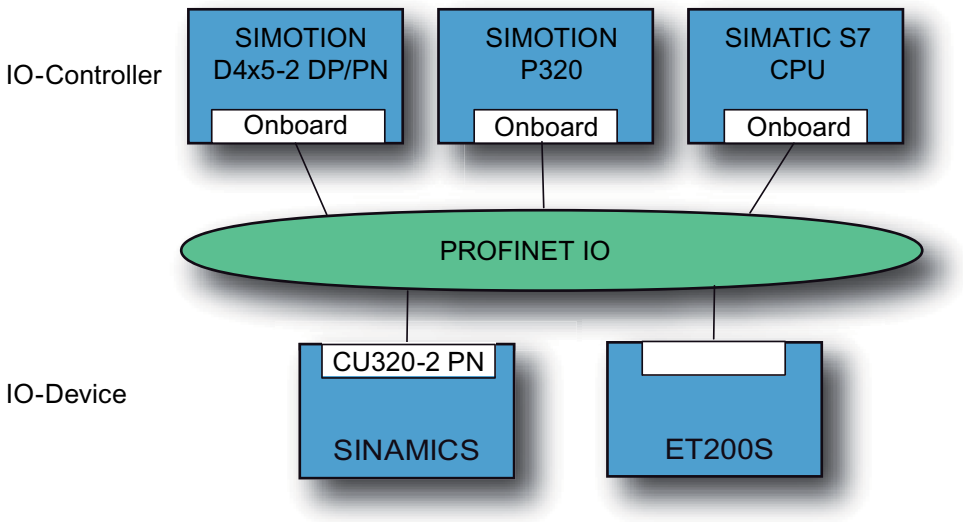


Bild 4-1 Beispiele für IO-Controller und IO-Devices

4.1.4 IO-Device

Dezentrale Feldgeräte wie I/O-Komponenten (z. B. ET200) oder Antriebe (z. B. SINAMICS S120 mit CU320-2 PN) werden als IO-Device bezeichnet. Die Funktion ist mit einem PROFIBUS DP Slave vergleichbar.

Siehe auch

IO-Device anlegen (Seite 101)

4.1.5 PROFINET IO-System

Ein Controller und die ihm zugeordneten Devices bilden zusammen ein PROFINET IO-System.

4.1.6 I-Device

Das I-Device bei PROFINET ist vergleichbar mit der Funktion des I-Slaves bei PROFIBUS, d. h. eine SIMOTION CPU kann die Rolle eines IO-Device übernehmen und so mit einem anderen IO-Controller Daten austauschen.

Während bei PROFIBUS eine Schnittstelle entweder nur Master oder nur Slave sein kann, ist es bei PROFINET möglich, auf einer PROFINET-Schnittstelle IO-Controller und IO-Device gleichzeitig zu sein.

4.1.7 RT-Klassen

4.1.7.1 RT-Klassen bei PROFINET IO

Beschreibung

PROFINET basiert auf dem Ethernet Standard. Daher können alle auf Ethernet basierenden Standard-Protokolle (z. B. HTTP, FTP, TCP, UDP, IP ...) über das PROFINET-Netzwerk übertragen werden.

Neben den aus der Office-Welt Welt bekannten Protokollen bietet PROFINET zwei an die Anforderungen der Automatisierung angepasste Protokolle (Übertragungsarten) an. Es handelt sich um PROFINET IO mit RT und IRT.

Die beiden Übertragungsarten sind für die zyklische IO-Kommunikation mit geringen Datenmengen innerhalb eines Netzwerkes optimiert.

RT

Die RT-Kommunikation nutzt die im Ethernet Standard beschriebene Möglichkeit der Priorisierung von Telegrammen. Dieser Mechanismus wird z. B. auch bei Voice over IP genutzt. Für detaillierte Informationen, siehe PROFINET IO mit RT (Seite 46).

IRT

Bei PROFINET IO mit IRT wird dem Ethernet ein Zeitschlitzverfahren überlagert. D. h., es entstehen 2 Slots, im ersten Slot werden die IRT-Telegramme, im zweiten Slot die RT und IP-Telegramme übertragen. Damit wird für die IRT-Daten Übertragungsbandbreite reserviert, die bei allen Last/Überlastsituationen garantiert wird. Damit alle beteiligten Geräte wissen, wann der Zeitschlitz beginnt, setzt IRT die Synchronisierung der Geräte voraus.

In der Übertragungsart IRT wird zwischen den beiden Realtime Klassen Hohe Flexibilität und Hohe Performance unterschieden.

IRT - Hohe Flexibilität

Die Realtimeklasse IRT Hohe Flexibilität entspricht der zuvor beschriebenen Übertragungsart IRT. Es wird ein für das gesamte Netzwerk einheitlicher IRT-Slot im Engineering definiert.

IRT - Hohe Performance

Neben der Bandbreitenreservierung erfolgt eine zeitliche Planung der zyklischen Telegramme unter Berücksichtigung der Topologie. Dadurch kann vom Engineering für jedes Kabel individuell die benötigte Bandbreite ermittelt werden. Damit kann gegenüber IRT Hohe Flexibilität das IRT-Zeitintervall minimiert und somit die Übertragung optimiert werden.

Neben der Synchronisation des Übertragungsnetzes bei IRT können bei IRT Hohe Performance auch die Applikation (z. B. Lageregler und Interpolator der SIMOTION) in den Geräten synchronisiert werden (taktsynchrone Applikation). Dies entspricht dem Verhalten des taktsynchronen PROFIBUS.

Das ist unbedingte Voraussetzung für das Schließen von Regelkreisen über das Netzwerk und taktsynchrones Schalten von Ein- und Ausgängen im Netz.

Hinweis

Für SIMOTION wird ausschließlich IRT Hohe Performance verwendet. Wird im weiteren Dokument von IRT gesprochen, ist damit IRT Hohe Performance gemeint.

Für detaillierte Informationen, siehe PROFINET IO mit IRT (Hohe Performance) (Seite 48).

RT und IRT im Vergleich

Tabelle 4- 1 Die wichtigsten Unterschiede zwischen RT und IRT

Eigenschaft	RT	IRT (Hohe Flexibilität)	IRT (Hohe Performance)
Realtimeklasse	Realtime Class 1	Realtime Class 2	Realtime Class 3
Übertragungsart	Priorisierung der zyklischen RT-Daten durch Ethernet-Prio (VLAN-Tag)	Bandbreitenreservierung, d. h. Reservierung eines Zeitbereichs, in dem nur zyklische IRT Daten, aber keine RT oder IP-Telegramme übertragen werden.	Vom Engineering optimierte Bandbreitenreservierung auf Basis von Topologieinformationen.
Determinismus	Varianz der Übertragungsdauer der zyklischen RT-Daten durch TCP/IP-Telegramme	Garantierte Übertragung der zyklischen IRT-Daten innerhalb des reservierten IRT-Zeitintervalls	Sende- und Empfangszeitpunkte der zyklischen IRT-Daten liegen exakt fest und sind garantiert für beliebige Topologien
taktsynchrone Applikation	nicht unterstützt	nicht unterstützt	unterstützt
Hardwareunterstützung durch spezielle Ethernet-Controller	Nein	Ja	Ja

4.1.7.2 Sendetakt und Aktualisierungszeit

Beschreibung

Im PROFINET-System werden die beiden Takte Sendetakt und Aktualisierungszeit unterschieden. Der Sendetakt ist der Grundtakt für die zyklische Kommunikation. Die Aktualisierungszeit gibt an, in welchem Zyklus ein Device mit Daten versorgt wird.

Sendetakt

Ist der Zeitraum zwischen zwei aufeinander folgenden Intervallen für IRT- bzw. RT-Kommunikation. Der Sendetakt ist das kleinstmögliche Sende-Intervall für den Datenaustausch. Der Sendetakt entspricht der kleinstmöglichen Aktualisierungszeit. Innerhalb dieser Zeit werden IRT-Daten und Nicht-IRTDaten (RT, TCP/IP) übertragen. Alle Geräte einer Sync-Domain arbeiten mit dem gleichen Sendetakt.

Aktualisierungszeit

Die Aktualisierungszeit kann für jedes IO-Device separat projektiert werden und bestimmt den Zeitabstand, in dem Daten vom IO-Controller zum IO-Device (Ausgänge) sowie Daten vom IO-Device zum IO-Controller (Eingänge) gesendet werden. Die berechneten/projektierten Aktualisierungszeiten sind immer Vielfache (2^n) des Sendetaktes.

Zusammenhang zwischen Aktualisierungszeit und Sendetakt

Die berechneten Aktualisierungszeiten sind Vielfache (1, 2, 4, 8, ..., 512) des Sendetakts. Die minimal erreichbare Aktualisierungszeit ist damit abhängig vom minimal einstellbaren Sendetakt des IO-Controllers und der Leistungsfähigkeit des IO-Controllers und des IO-Devices.

4.1.7.3 Einstellbare Sendetakte und Aktualisierungszeiten

Beschreibung

Die nachfolgende Tabelle beschreibt die bei PROFINET IO für SIMOTION-Geräte einstellbaren Sendetakte und die davon abhängig einstellbaren Untersetzungen für IRT und RT. Die einstellbaren Sendetakte sind in zwei Bereiche gegliedert: Bereich "gerade" und Bereich "ungerade". Aktualisierungszeiten ergeben sich aus dem Produkt von Untersetzungen und Sendetakt.

Hinweis

Die folgenden Ausführungen beziehen sich auf die Verwendung des 1. Servo-Takts. Bei der Verwendung von Servo_fast bzw. IPO_fast gelten unter Umständen andere Restriktionen. Wird der Servo_fast und IPO_fast verwendet, dann muss der PROFINET taktsynchron betrieben werden.

Tabelle 4- 2 Einstellbare Sendetakte und Aktualisierungszeiten bei Verwendung von 1. Servo

Sendetakt		**Untersetzung (Aktualisierungszeit = Faktor * Sendetakt)**	
		RT	**IRT Hohe Performance**
Bereich "gerade"	250, 500, 1000 µs	1,2,4,8,16,32,64,128,256,512	1 *Hinweis 2)*
	2000 µs	1,2,4,8,16,32,64,128,256	
	4000 µs	1,2,4,8,16,32,64,128	
Bereich "ungerade" *Hinweis 1*)	375, 625, 750, 875, 1125, 1250 µs ... 3875 µs (Schrittweite 125 µs)	nicht unterstützt	1

Wenn in einem PROFINET IO-System kein Sync-Master projektiert ist (kein PROFINET IRT), dann kann der Sendetakt für das betreffende PROFINET IO-System individuell am betreffenden IO-Controller in den Eigenschaften **<PROFINET-Schnittstelle>** im Reiter PROFINET unter Sendetakt oder in den Eigenschaften **PROFINET IO-System** im Reiter Aktualisierungszeit eingestellt werden. Als Default-Sendetakt ist 1 ms eingestellt. Im Register IO-Zyklus können über den Modus fixierter Faktor oder fixierte Aktualisierungszeit und den Faktor die Untersetzung für die Aktualisierungszeit eingestellt werden.

Sobald ein Sync-Master im PROFINET IO-System projektiert ist, wird der Sendetakt unter den Eigenschaften der Sync-Domain definiert. Die der Sync-Domain zugeordneten Controller übernehmen diesen Wert. Aktualisierungszeiten können für jedes IO-Device unabhängig voneinander eingestellt werden.

Hinweis 1) Mischbetrieb RT / IRT Hohe Performance
Ungerade Sendetakte können nur verwendet werden, wenn sich in den an der Sync-Domain beteiligten IO-Systemen kein RT IO-Device befindet. Wenn sich IO-Devices mit RT-Klasse "RT" in einer Sync-Domain befinden, dann können nur noch die Sendetakte aus dem Bereich "gerade" eingestellt werden.

Hinweis 2) Untersetzung (Faktor) und taktsynchrone Applikation
Einige IO-Devices unterstützen mit IRT hohe Performance neben Faktor 1 weitere Faktoren 2, 4, 8, 16.

Wenn IO-Devices (z. B. ET200S IM151-3 PN HS, SINAMICS S) mit taktsynchroner Applikation betrieben werden, kann generell nur der Faktor 1 eingestellt werden.
Der Modus für die Aktualisierungszeit ist dabei immer auf **fixierter Faktor** zu stellen, damit STEP 7 keine automatische Anpassung der Aktualisierungszeit vornimmt und diese damit immer dem Sendetakt entspricht.

Modus Sendetakt zur Aktualisierungszeit

- fixierter Faktor
  feste Untersetzung Sendetakt zur Aktualisierungszeit
- fixierte Aktualisierungszeit
  Aktualisierungszeit wird eingestellt
- automatisch
  STEP 7 passt die Untersetzung automatisch an, falls der Faktor zu klein gewählt wurde

Hinweis

Es wird empfohlen mit der Einstellung **fixierter Faktor** zu arbeiten.

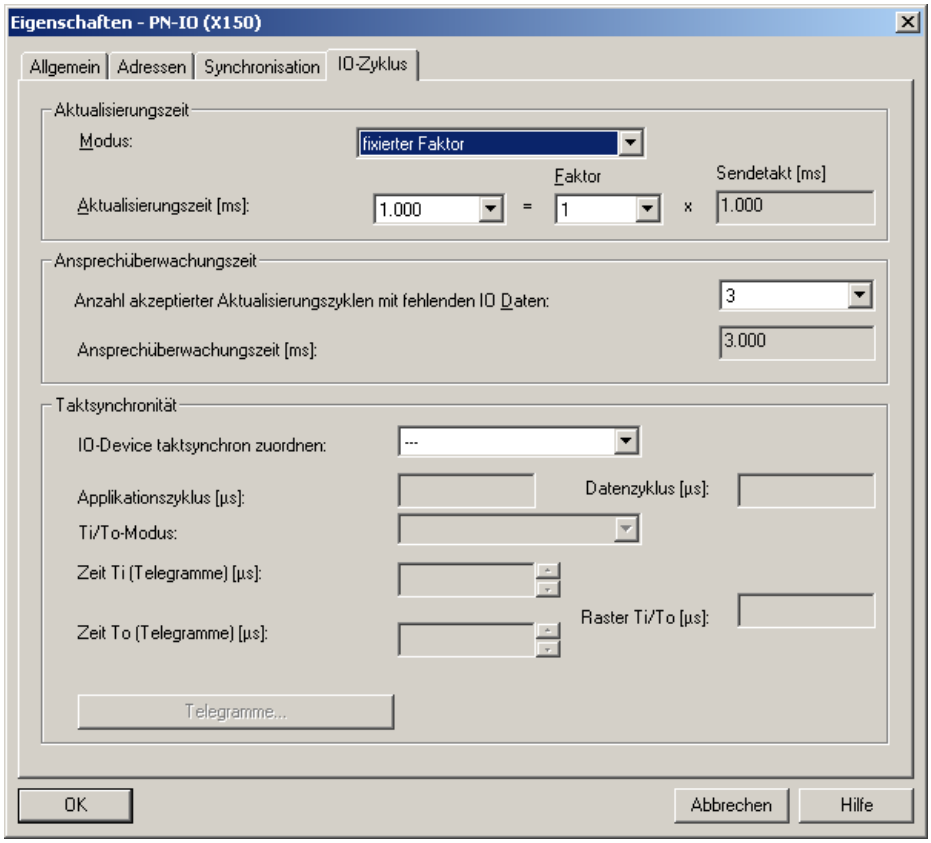


Bild 4-2 Aktualisierungszeit und Faktor

4.1.7.4 RT-Klassen einstellen

RT-Klassen

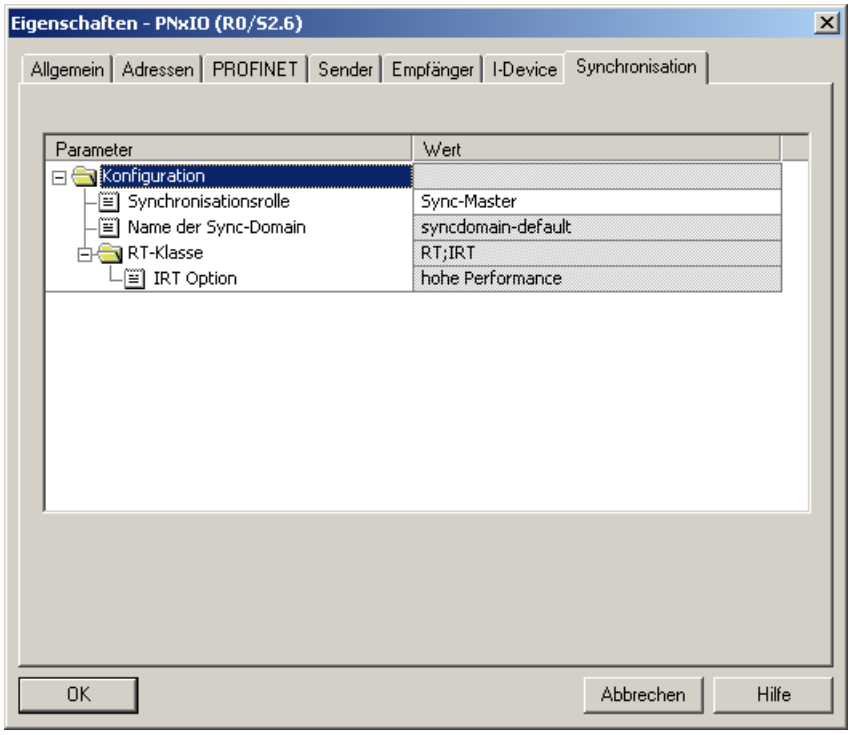
Der IO-Controller bestimmt, welche RT-Klasse sein IO-System unterstützt, indem an seiner Controllerschnittstelle die Realtimeklasse eingestellt wird. RT-Devices können immer betrieben werden, auch wenn IRT-Klassen eingestellt sind.

Einstellen der RT-Klasse

Die RT-Klasse können Sie in HW Konfig für das jeweilige PROFINET-Gerät einstellen.

1. Doppelklicken Sie in HW Konfig auf den Eintrag der PROFINET-Schnittstelle in der Baugruppe.

   Der Dialog **Eigenschaften** wird aufgerufen.



2. Wählen Sie in der Registerkarte **Synchronisation** unter RT-Klasse die Echtzeitklasse aus. Bei IRT wird automatisch IRT Hohe Performance bei SIMOTION gesetzt.
3. Bestätigen Sie mit **OK**.

Hinweis

Für Motion Control Applikationen mit SIMOTION und SINAMICS wird ausschließlich IRT Hohe Performance verwendet.

4.1.7.5 PROFINET IO mit RT

PROFINET IO mit RT ist die optimale Lösung für die Einbindung von Peripheriesystemen ohne besondere Anforderungen an Performance und Taktsynchronität. Es handelt sich hierbei um eine Lösung, die auf Standard-Ethernet IC (Ethernet Controller) und handelsüblichen Industrial Switches als Infrastruktur-Komponenten aufsetzt. Eine spezielle Hardware-Unterstützung ist nicht erforderlich.

Nicht taktsynchron

Standard-Ethernet und PROFINET IO mit RT bietet keine Synchronisationsmechanismen für Geräte, aber unterbinden diese Möglichkeit auch nicht. Es ist daher keine taktsynchrone Datenübertragung möglich und damit auch keine taktsynchrone Applikation für Motion Control.

Datenaustausch

Die Kommunikation über PROFINET IO mit RT und IRT basiert auf dem Ethernetframe und der MAC-Adresse. Daher ist eine netzwerkübergreifende Kommunikation über Router hinweg mit RT und IRT nicht möglich. PROFINET IO Telegramme werden gemäß IEEE802.1Q gegenüber IT-Telegrammen priorisiert. Damit sind die in der Automatisierungstechnik erforderlichen Echtzeiteigenschaften z. B. für Standard IOs erfüllt.

Aktualisierungszeit

Die einstellbare Aktualisierungszeit liegt im Bereich von 0.25 - 512 ms. Die gewählte Aktualisierungszeit ist von den Prozessanforderungen, von der Anzahl der Geräte und der Anzahl der IO-Daten abhängig Aufgrund der Performancesteigerung bei PROFINET gegenüber Feldbussen ist der Buszyklus in der Regel nicht mehr die den Systemzyklus bestimmende Größe.

4.1.7.6 PROFINET IO mit IRT - Überblick

Überblick

Mit PROFINET IO mit IRT werden die Kommunikationsanforderungen die über Standard Signale hinaus gehen erfüllt. Mit IRT wird der bei RT noch mögliche Jitter in der Kommunikation durch die Synchronisation des Netzwerks signifikant reduziert.

Dazu wird dem Ethernet Netzwerk ein Zeitschlitzverfahren überlagert. Es wird ein Zeitschlitz für die IRT-Telegramme und einen für die RT und IP-basierten Telegramme reserviert. Voraussetzung für ein solches Verfahren ist die Synchronität aller an der IRT-Kommunikation beteiligten Geräte.

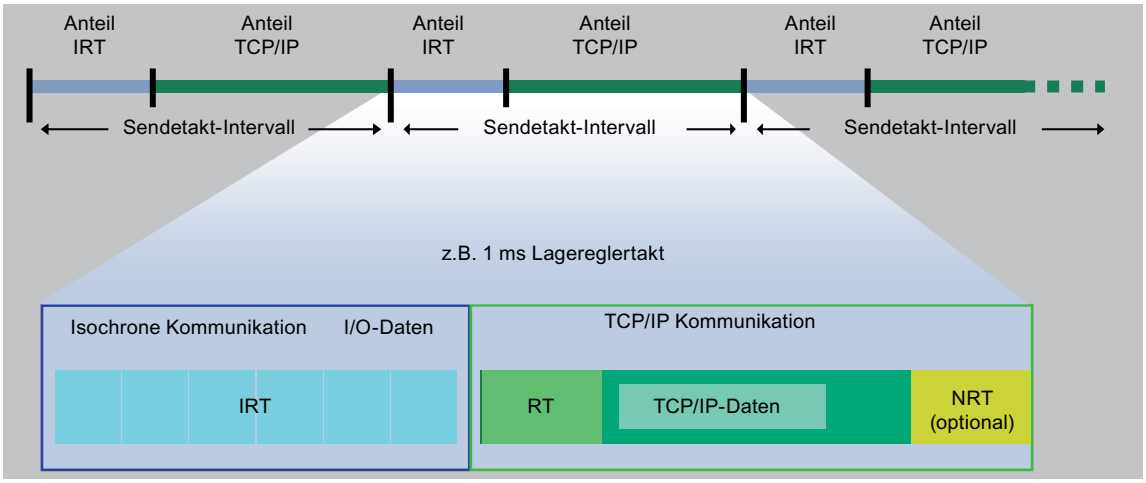


Bild 4-3 IRT Kommunikation - Überblick

PROFINET IO mit IRT gibt es in zwei Ausprägungen:

- IRT Hohe Flexibilität mit fester Bandbreitenreservierung
- IRT Hohe Performance (Seite 48) mit optimierter Bandbreitenreservierung und geplanter IRT-Kommunikation

Bei PROFINET IO mit IRT werden alle IRT-Geräte auf einen gemeinsamen Sync-Master synchronisiert. Siehe auch Äquidistanz und Taktsynchronität bei PROFINET (Seite 50) .

4.1.7.7 PROFINET IO mit IRT (Hohe Performance)

Für Motion Control Anwendungen wird die Leistungsfähigkeit mit PROFINET IO IRT (Hohe Performance) deutlich erweitert. Beim Einsatz von Feldbussen z. B. PROFIBUS werden die Geräte parallel an den Bus angeschlossen. Dies hat Konsequenzen, erstens es kann zu einem Zeitpunkt immer nur 1 Gerät senden. Zweitens ist aufgrund der Parallelschaltung aller Geräte mit ca. 12 Mbit/s die physikalische Grenze erreicht.

PROFINET setzt auf die Ethernet-Technologie, die Punkt zu Punkt Verbindungen vorsieht. Bei Punkt zu Punkt Verbindungen kann die Übertragungsrate gegenüber der Parallelverdrahtung erheblich gesteigert werden. Bei PROFINET werden 100 MBit/s genutzt. In Verbindung mit der Switch-Technologie werden alle Verbindungskabel voneinander entkoppelt, d. h. gleichzeitig kann auf jedem Kabel gesendet und empfangen werden.

Durch die zeitliche Planung des Telegrammverkehrs bei IRT Hohe Performance erreicht man eine wesentliche Optimierung des Datenverkehrs, da nur die wirklich notwendige Bandbreite reserviert wird.

IRT (Hohe Performance) eignet sich besonders für:

- die Regelung und Synchronisation von Achsen über PROFINET IO
- eine schnelle, taktsynchrone Peripherieeinbindung mit kurzen Klemme-Klemme-Zeiten

Sendetakt

Der Sendetakt kann bei PROFINET IRT Hohe Performance von 250 µs bis 4 ms eingestellt werden. Im Mischbetrieb RT und IRT Hohe Performance sind nur die Werte 0.5, 1.0, 2.0 oder 4.0 einstellbar.

Der tatsächlich genutzte Sendetakt ist von verschiedenen Faktoren abhängig:

- vom Prozess, es sollte nur so schnell wie nötig kommuniziert werden. Dies reduziert Buslast und CPU Belastung.
- der Busauslastung (Anzahl der Geräte und IO-Daten pro Gerät)
- in der Steuerung zur Verfügung stehende Rechenleistung
- unterstützte Sendetakte in den beteiligten PROFINET-Geräten einer Sync-Domain.

Ein typischer Sendetakt ist z. B. 1 ms; Er kann aber in einem Raster von 125 µs in den Grenzen von 250 µs bis 4 ms eingestellt werden. Siehe auch Einstellbare Sendetakte und Aktualisierungszeiten (Seite 43).

Die unterstützten Sendetakte entnehmen Sie bitte den entsprechenden Handbüchern der jeweiligen SIMOTION-Geräte. Eine minimale Zykluszeit von 250 µs wird nur von ausgewählten Komponenten unterstützt (SIMOTION P320-3, P350-3, D445-2 DP/PN, D455-2 DP/PN und ET 200S HS Baugruppen).

Taktsynchrone Applikation

Taktsynchrone Datenübertragung und eine auf das Bussystem synchronisierte Applikation erfüllen Anforderungen für anspruchsvolle Motion Control Anwendungen. Damit ist es möglich Regelkreise über das Bussystem zu schließen und minimale garantierte Reaktionszeiten (Klemme-Klemme-Zeitverhalten) zu erreichen. Darüber hinaus wird eine performante und taktsynchrone Anbindung an die Applikation mit geringer Belastung der Applikations-CPU sichergestellt.

Im Gegensatz zu Standard Ethernet und PROFINET IO mit RT werden die Telegramme bei PROFINET IO mit IRT Hohe Performance zeitlich geplant übertragen.

Zeitlich geplante Datenübertragung

Unter zeitlicher Planung versteht man die Festlegung der Kommunikationspfade und die exakten Übertragungszeitpunkte für die zu übertragende Daten. Durch eine Kommunikationsplanung wird die Bandbreite optimal ausgenutzt und damit eine bestmögliche Performance erzielt. Dafür muss die Topologie des Netzwerks projektiert werden und aus der projektieren Topologie berechnet das Engineering-System automatisch die Kommunikationsplanung (Siehe auch Topologie). Die für PROFINET IO relevanten Daten werden durch einen Download von HW Konfig in den IO-Controller übertragen. Durch die zeitliche Festlegung der Übertragungszeitpunkte wird die höchste Determinismus-Qualität erreicht, die vor allem für eine taktsynchrone Applikationsanbindung vorteilhaft ist.

Datenaustausch

PROFINET IO mit IRT Hohe Performance und PROFINET IO mit IRT Hohe Flexibilität läuft nur innerhalb einer Sync-Domain. Diese dürfen jedoch nicht in einem IO-System gemischt werden. D. h., eine Sync-Domain kann aus 2 oder mehr IO-Systemen bestehen, die alle zueinander synchronisiert werden können. Innerhalb des IO-Systems wird dann entweder IRT Hohe Flexibilität oder IRT Hohe Performance verwendet.

4.1.8 Sync-Domain

Eine Sync-Domain ist eine Gruppe von PROFINET-Geräten, die auf einen gemeinsamen Takt synchronisiert sind. Der Sync-Master gibt den Takt vor. Der Sync-Slave synchronisiert sich auf den vom Sync-Master vorgegebenen Takt. Eine Sync-Domain hat einen Sync-Master.

Siehe auch

Sync-Domain anlegen (Seite 89)

4.1.9 Äquidistanz und Taktsynchronität bei PROFINET

Beschreibung

PROFINET IO mit IRT basiert auf Ethernet mit überlagertem Zeitschlitzverfahren. Dafür ist die Synchronisation aller an der Kommunikation beteiligten Busanschaltungen Voraussetzung. Bei PROFINET IO mit IRT Hohe Performance und IRT Hohe Flexibilität überträgt ein Sync-Master ein Synchronisationstelegramm, auf das sich alle Sync-Slaves aufsynchronisieren.

Sync-Master, -Slaves und -Domain

Die Geräterollen Sync-Master und Sync-Slave werden bei der Projektierung zugewiesen. Die Rolle eines Sync-Masters kann einem IO-Controller oder SCALANCE 200 IRT-Switches zugewiesen werden.

Eine Sync-Domain kann aus PROFINET Geräten mit IRT Hohe Performance und aus PROFINET Geräten mit IRT Hohe Flexibilität bestehen. An den Enden (Stichleitungen) können sich auch PROFINET-Geräte mit RT befinden, nicht aber zwischen zwei Geräten mit PROFINET IO mit IRT (Hohe Flexibilität oder Hohe Performance).

Eine Linie (Netzwerk) darf kein IRT Hohe Flexibilität oder IRT Hohe Performance gemischt werden, denn diese müssen immer direkt miteinander verbunden sein.

Kompatibilität

Die Kommunikation zwischen und durch verschiedene Sync-Domains hindurch ist über PROFINET IO mit RT möglich.

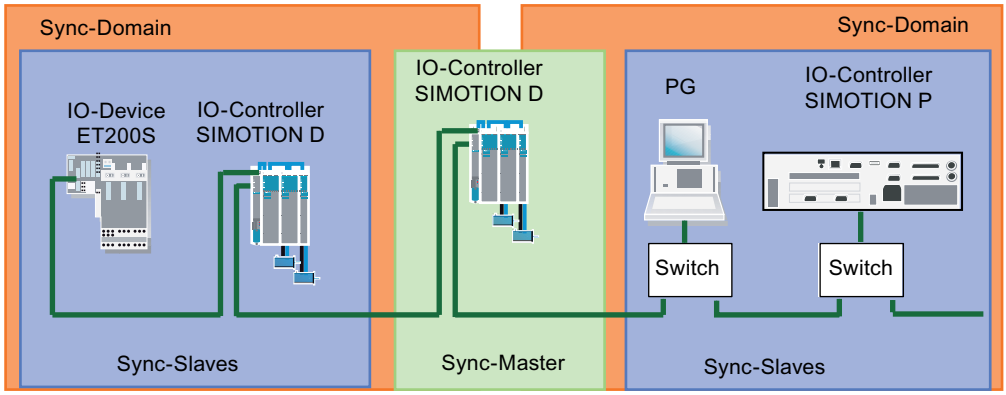


Bild 4-4 Taktsynchronität PROFINET

In dem Aufbau werden IRT-fähige Switche z. B SCALANCE X204 IRT verwendet. Alle an der IRT-Kommunikation beteiligten Geräte sind direkt miteinander verbunden. Das PG in der Mitte des Netzwerkes ist über eine Stichleitung angeschlossen und unterbricht daher nicht die IRT-Strecke.

Siehe auch

Taktsynchrone Applikationen bei PROFINET (Seite 57)

4.1.10 Adressierung von PROFINET IO Geräten

Für den Datenaustausch über Ethernet wird eine weltweit eindeutige Adresse, die MAC (Media Access Control)-Adresse verwendet, die Bestandteil des Ethernet-Telegramms ist. Die MAC-Adresse ist an die Hardware gebunden und kann nicht verändert werden.

Auf Ethernet aufsetzende Protokolle wie z. B. HTTP (Webanwendungen) oder FTP (Filetransfer) nutzen das IP-Protokoll. Die Adressierung erfolgt über die IP-Adresse. Dies ist eine logische Adresse, die vom Anwender vergeben werden kann.

PROFINET verwendet, neben den beiden im Zusammenhang mit Ethernet bereits bekannten Adressinformationen, zur Identifizierung der PROFINET-Geräte zusätzlich einen Gerätenamen (NameOfStation). Bei dem Gerätenamen handelt es sich um einen String, der den Anforderungen eines DNS (Domain Name Service)-Namens genügt. Dieser Gerätename, auch Kommunikationsname genannt, muss eindeutig im PROFINET-Netzwerk sein.

In der Inbetriebnahme-Phase wird jedem PROFINET-Gerät (identifiziert über MAC-Adresse) durch das Projektierungswerkzeug einmalig ein Gerätename zugewiesen und dieser im PROFINET-Gerät remanent gespeichert (sog. Knotentaufe). Über den Gerätenamen wird ein Gerät in der Projektierung referenziert. Wird ein Gerät, z. B. aufgrund eines Defektes ausgetauscht, so verfügt das neue Gerät über eine andere MAC-Adresse. Wird es mit dem Gerätenamen des ausgetauschten Gerätes getauft (z. B. durch Umstecken eines Wechselmediums, das den Gerätenamen remanent speichert), kann es ohne Änderungen in der Projektierung die Funktion des getauschten Gerätes übernehmen.

Alternativ kann die Taufe der Devices anhand von Topologieinformationen automatisch vom Controller durchgeführt werden. Dazu ist es notwendig, dass im Engineeringsystem Topologieinformationen (wer ist wie mit wem verdrahtet) hinterlegt werden. Im Hochlauf identifiziert der Controller die angeschlossenen Devices über den Gerätenamen und weist dann dem Device die im Engineering definierte IP-Adresse zu. Damit ist die Station über IP-Dienste erreichbar. Die IP-Adresse kann einem projektierten Nummernband entnommen werden oder individuell projektiert sein.

Ein PROFINET-Gerät hat folgende Adressen, über das es angesprochen werden kann:

- MAC-Adresse (Bestandteil Ethernet-Telegramm, wird auf Gerät gespeichert und ist nicht veränderbar)
- IP-Adresse (IP-basierte Kommunikation, z. B. Engineering-Zugriffe, muss allen Geräten zugewiesen werden)
- Gerätenamen, Kommunikationsnamen (Identifikation der Geräte im Hochlauf durch Controller)

Siehe auch

Gerätenamen und IP-Adressen für IO-Devices vergeben (Seite 108)

4.1.11 Planung und Topologie für ein PROFINET Netzwerk

Planungsrichtlinie

Für die Auslegung eines PROFINET Netzwerkes müssen vor der Realisierung grundlegende Fragen geklärt werden. In diesem Kapitel erhalten Sie einen groben Leitfaden, der Sie bei der Definition der Anforderungen und Erstellung von Planungsunterlagen unterstützt. Die Planung ist ein iterativer Prozess, d. h. einige Anforderungen beeinflussen sich gegenseitig bedingen daher Änderungen bei der Gesamtplanung.

Inhalt der Planungsunterlagen

Nach der Festlegung der Anforderungen sollten Ihnen am Ende des Planungsprozess folgenden Informationen vorliegen:

- Anlagenaufbau
- Topologie

- Komponentenauswahl
- Auswahl des Übertragungsmediums
- Steckverbinder-Auswahl
- Kommunikationsbeziehungen
- Abschätzung über die zu übertragenden Datenmengen

Vorüberlegungen und Analyse während der Planung

1. **Auswahl der Geräte**

   Erstellen Sie eine Geräteliste. Die Auswahl der Geräte richtet sich unter anderem nach der Applikationsklasse (Conformance-Class), der Zeit- und Kommunikationsanforderungen, der Funktion sowie Umgebungseinflüsse und Schutzart.

2. **Position der Geräte in der Maschine**

   Aus der Position der Geräte in der Maschine ergeben sich Zusammenhänge bezüglich der Schutzklasse, EMV, Geräteabmessungen und der verwendeten Leitungen z. B. Lichtwellenleiter anstatt Kupferkabel. Dies wiederum beeinflusst die Geräteauswahl.

3. **Definition der Kommunikationseigenschaften**

   Die Zeitanforderungen bezüglich der Applikation (taktsynchron/zyklisch) und der Kommunikation über PROFINET müssen definiert sein, d. h. erfolgt die Kommunikation in Echtzeit (IRT/RT) oder azyklisch über TCP/IP oder UDP/IP.

4. **Planung des Netzwerkes (Topologie)**

   Legen Sie die Topologie (Ring, Stern, Linie) des Netzwerks fest. Abhängig von der Topologie müssen Switche (IRT-fähig), EMV, Ausdehnung des Netzwerkes, WLAN (kein IRT möglich) und eventuelle Medienredundanz MRP (nicht möglich bei SIMOTION/SINAMICS) berücksichtigt werden.

   – Bauen Sie Ihr PROFINET-System, wo sinnvoll, sternförmig auf (z. B.: verzweigen Sie nach einer CPU über ein Switch auf eine sternförmige Topologie).

   – Bei PROFINET mit IRT ist der Aufbau einer Linie mit 64 IRT-Geräten zulässig. Abhängigkeiten bestehen von den übertragenen Datenmengen. Werden größere Telegrammlängen pro Device projektiert, kann sich die mögliche Anzahl pro Linie reduzieren. Dies wird schon vorzeitig bei der Projektierung mit HW Konfig erkannt und mit einer Fehlermeldung signalisiert. Die 64 IRT-Geräte in Linie gelten nur für PROFINET nach V2.2.

   – Halten Sie die Verkettungstiefe der Switches gering. Für die RT-Kommunikation wird damit der Worst case Jitter reduziert. Bei IRT Hohe Flexibilität wird der vom Engineering ermittelte Bandbreitenbedarf reduziert.

Topologie	
Stern	Durch den Anschluss von Kommunikationsteilnehmern an einen Switch entsteht automatisch eine sternförmige Netztopologie. Wenn ein einzelnes PROFINET-Gerät ausfällt, führt das bei dieser Struktur im Gegensatz zu anderen Strukturen nicht zwangsläufig zum Ausfall des gesamten Netzes. Lediglich der Ausfall eines Switches führt zum Ausfall der dahinter liegenden Geräte.
Baum	Wenn Sie mehrere sternförmige Strukturen miteinander verschalten, entsteht eine baumförmige Netztopologie.
Linie	Alle Kommunikationsteilnehmer werden in einer Linie hintereinander geschaltet. Wenn ein Switch ausfällt, dann ist eine Kommunikation über den ausgefallenen Switch hinweg nicht mehr möglich. Für die Realisierung der Linienstruktur müssen Geräte mit 2-Port-Switch verwendet werden. Der Aufwand für die Verkabelung ist bei einer linienförmigen Netzstruktur am geringsten.

5. **Definition der Kommunikationsbeziehungen (logische Zuordnung der Partner)**

   Legen Sie fest welche Kommunikationspartner in Verbindung stehen und wie die räumliche bzw. funktionale Zuordnung dieser Partner ist.

6. **Definition der Datenmengen.**

   Definieren Sie die möglichen Datenmengen der Netzwerkteilnehmer und an den Kommunikationsknotenpunkten.

   Hinweis

   Innerhalb eines PROFINET Netzwerkes sollte nur so schnell kommuniziert werden wie es technologisch für die Anlage erforderlich ist und nicht so schnell wie technisch möglich.

7. **Definition von Sendetakt und Aktualisierungszeit**

   Der Sendetakt richtet sich nach dem PROFINET-Gerät, dass die höchste Aktualisierung benötigt. Die Aktualisierungszeit ist ein Vielfaches des Sendetaktes und bestimmt die PROFINET-Netzlast. Die durch PROFINET erzeugte Netzlast sollte stets unterhalb von 50% liegen, um Reserven für Spitzenlasten zu besitzen. Berücksichtigen Sie, dass die PROFINET-Netzlast linear zu der Anzahl der PROFINET-Geräte und des Sendetakts ansteigt.

   Hinweis

   Projektieren Sie die Aktualisierungszeiten stets so, wie es der Prozess erfordert, auch wenn das Bussystem sehr viel kleinere Aktualisierungszeiten ermöglicht. Dies verringert die PROFINET-Netzlast und die Belastung des PROFINET-Controllers auf ein notwendiges Maß.

8. **Überprüfung der Netzlast**

   Zur Bestimmung der Netzlast müssen sowohl die PROFINET-Netzlast wie auch die Netzlast durch teilnehmende Standard-Ethernet-Geräte berücksichtigt werden. Dies können z. B. Videokameras zu Überwachung der Anlage oder Datenserver für Produktionsdaten sein. Ethernet-Geräte mit geringem Datenaufkommen wie z. B. Engineering-Arbeitsplätze oder HMI sind in der Regel unkritisch. Damit der PROFINET RT Datenstrom nicht beeinträchtigt wird, sollten Sie ggf. die Netzwerktopologie anpassen, wenn hohe Netzauslastungen von Ethernet-Teilnehmern zu erwarten sind. Berücksichtigen Sie aber auch ausreichend Bandbreitenreserve für zukünftige Erweiterungen.

   Hinweis

   Geräte mit hoher IP Last im Netzwerk möglichst in einem Bereich des Netzwerks separieren. Dies sind z. B. Diagnoseserver oder HMI Server.

9. **Anbindung Firmennetz**

   Falls die Automatisierungsanlage an das Firmennetz angebunden werden soll, müssen verschiedene Punkte beachtet werden. In der Regel sollte die Anbindung über einen Router oder einen SCALANCE S mit Firewall erfolgen, um unautorisierte Zugriffe zu verhindern. Eine PROFINET-IRT oder -RT Kommunikation ist über Router nicht möglich. Weitere Anbindungen z. B. Remote Access oder VPN sollten Sie im Einzelfall nur in Absprache mit der IT-Abteilung durchführen.

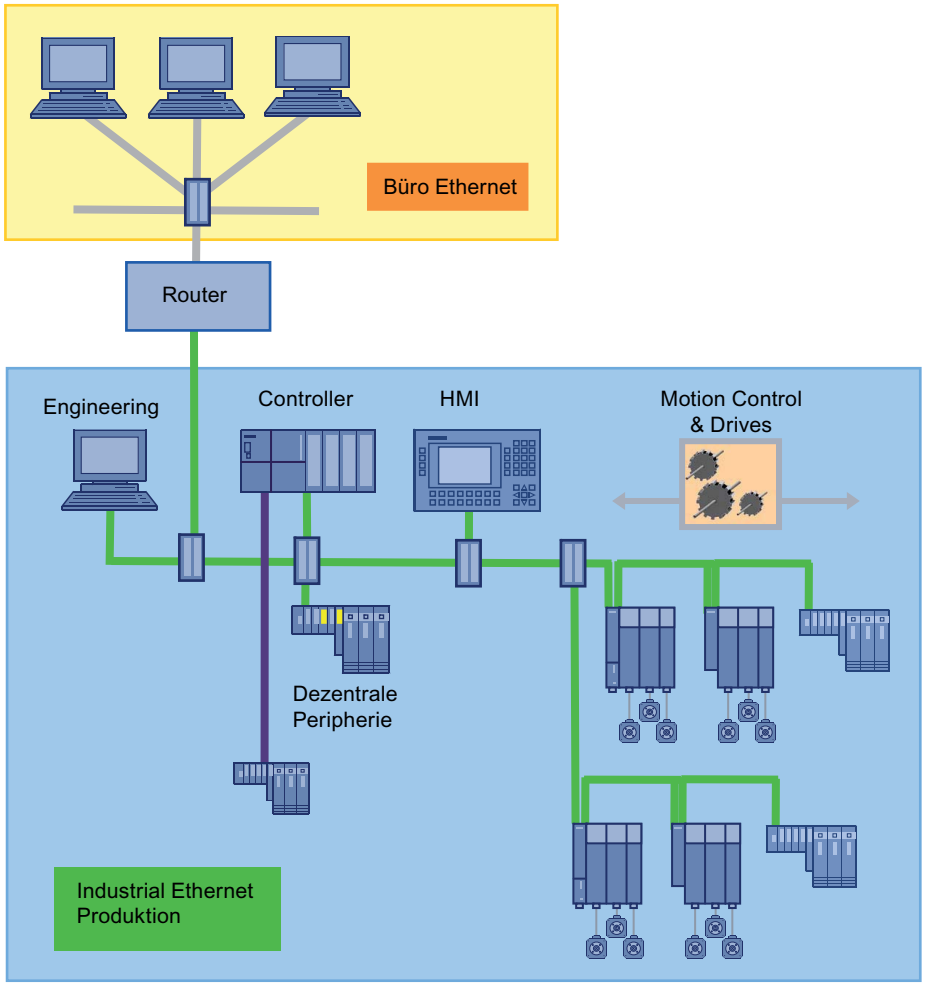


Bild 4-5 Optimierte Topologie einer Firmennetz-Anbindung

Projektierung der Topologie

Voraussetzung für die zeitliche Kommunikationsplanung ist die Kenntnis über die Netz-Topologie. Darunter versteht man die Informationen über das Zusammenschalten der einzelnen Geräte zu einem Kommunikationsnetz.

Die Topologieplanung ist nur für IRT Hohe Performance relevant. Die Netz-Topologie kann mithilfe eines in die Hardware Konfiguration integrierten Topologie-Editors benutzerfreundlich projektiert werden.

4.1.12 Taktsynchrone Applikationen bei PROFINET

Wie bei PROFIBUS kann auch bei PROFINET IO mit IRT Hohe Performance die Applikation auf den Takt des Übertragungsnetzes synchronisiert werden. Die Synchronisation aller PROFINET-Geräte mit IRT Hohe Performance auf eine gemeinsame Zeitbasis ist die Voraussetzung für die Realisierung verteilter Motion Control Applikationen und für Klemme/Klemme-Reaktionszeiten unter einer Millisekunde.

Hinweis

Die Taktsynchronität der Applikation zum Bus ist nur mit PROFINET IO mit IRT Hohe Performance möglich.

Projektierungsmodell Taktsynchronität ab V4.2

Das Projektierungsmodell der Taktsynchronität hat sich zu V4.2 verändert. Das Register **Applikation** bei den Eigenschaften des IO Devices wurde durch Einstellungen am Controller und im Register IO-Zyklus des IO Devices ersetzt. Eine ausführlichere Vorgehensweise zur Einstellung bei taktsynchronen Applikationen finden Sie im Kapitel SINAMICS S120 einfügen und projektieren (Seite 102)

Vorgehensweise

Zur Projektierung von taktsynchronen Applikationen gehen Sie wie im Folgenden beschrieben vor:

1. Stellen Sie bei Controller und Devices "IRT hohe Performance" ein, bei SIMOTION (Controller) ab V4.2 ist dies automatisch gesetzt.

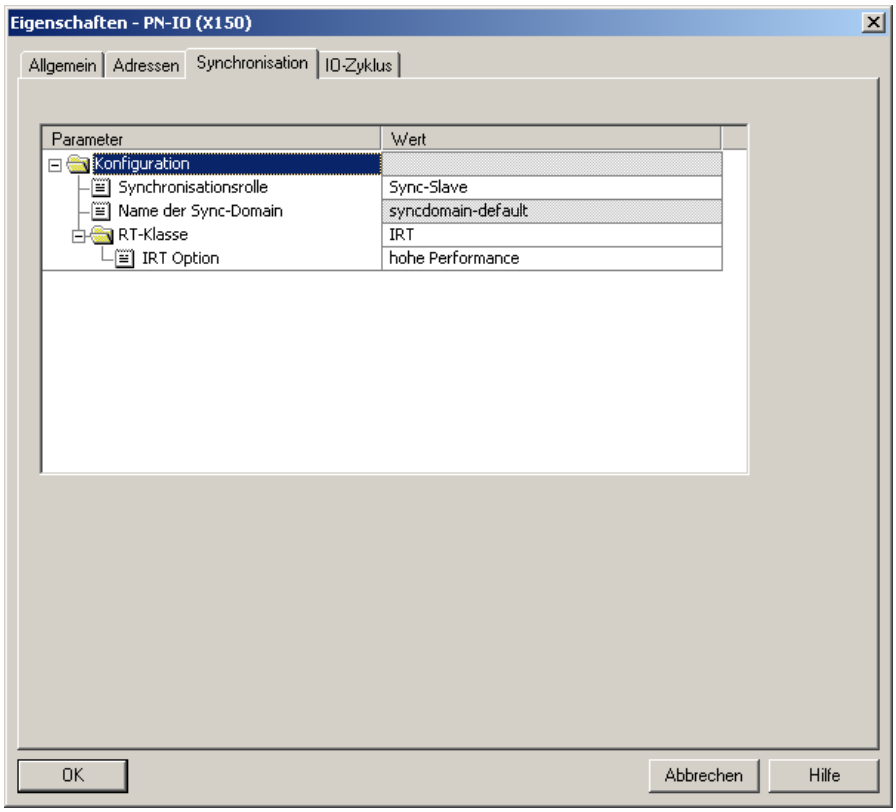
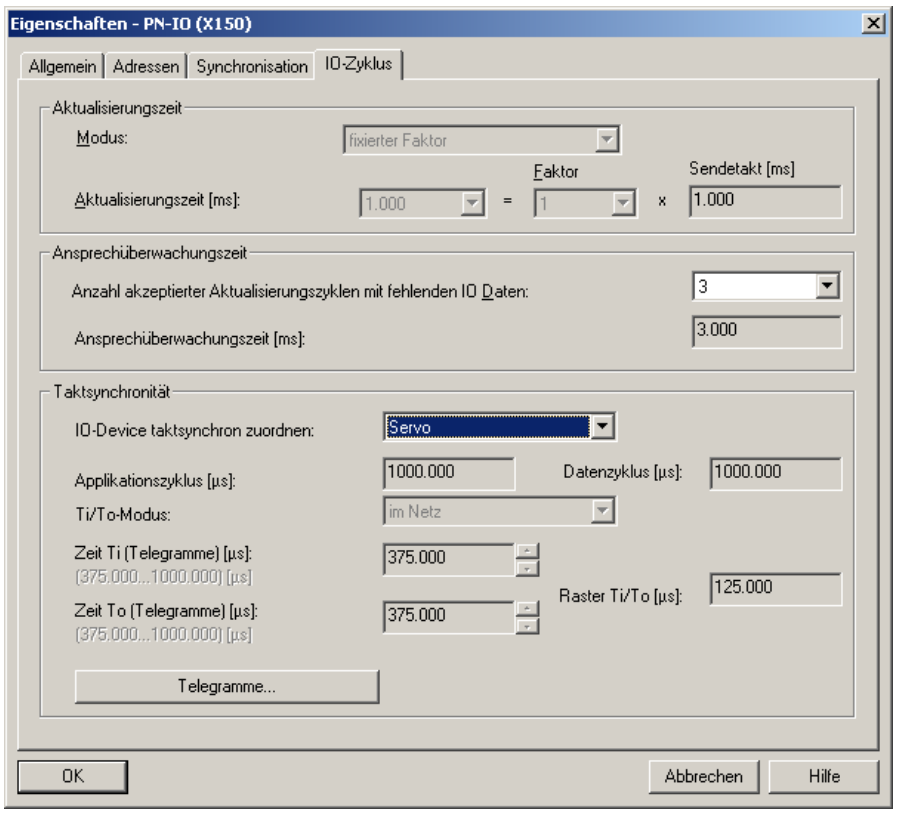


Bild 4-6 IO Device Synchronisation auf Sync-Slave setzen und IRT hohe Performance einstellen.

2. Stellen Sie bei den Devices den Modus **Aktualisierungszeit** auf **fixierter Faktor** und wählen Sie unter **IO-Device taktsynchron zuordnen** den Takt. In der Regel ist dies der Servo-Takt.



4.1.13 Azyklische Kommunikation über PROFINET

Beschreibung

Analog zu PROFIBUS DP ist es auch für PROFINET IO möglich, azyklische Kommunikation (Base Mode Parameter Access) zu betreiben. Eine ausführliche Beschreibung dazu finden Sie unter DP-V1 Kommunikation (Seite 282).

4.2 Eigenschaften von PROFINET IO mit SIMOTION

4.2.1 Einleitung

Voraussetzung

Um mit SIMOTION über PROFINET IO arbeiten zu können, müssen PN-Slots vorhanden sein. Diese sind direkt am Controller oder können mit dem Option Boards gesteckt werden.

Möglichkeiten der Anbindung:

- SIMOTION D4x5 mit Option Board CBE30
- SIMOTION D4x5-2 DP/PN
- SIMOTION P350 PN oder SIMOTION P320-3
- SIMOTION D410 PN
- SIMOTION C240 PN

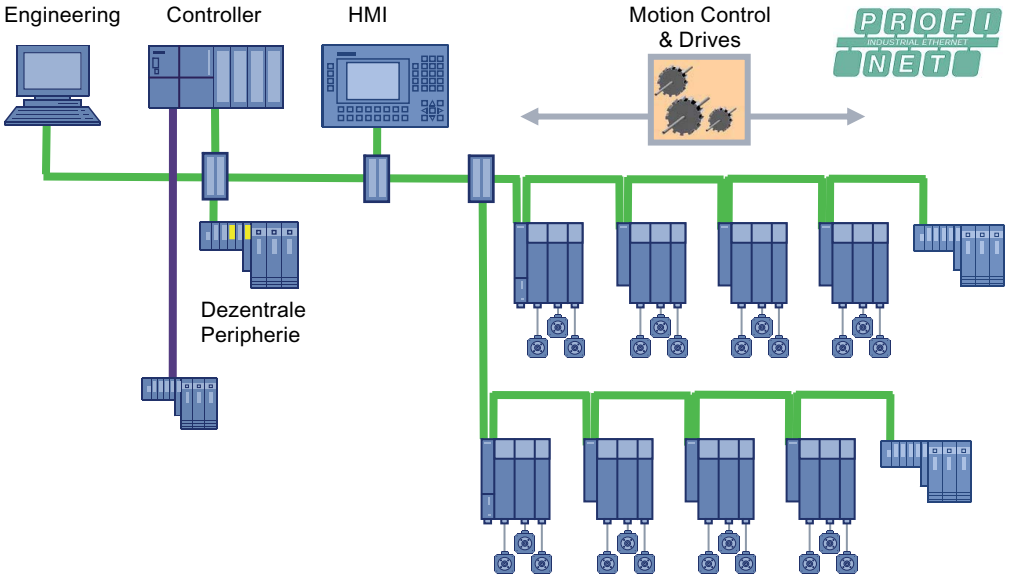


Bild 4-7 Anlagentopologie mit PROFINET

Die PROFINET-Geräte unterstützen den gleichzeitigen Betrieb von:

- IRT Hohe Performance - Isochronous Realtime Ethernet
  - Betrieb von IRT-Peripherie (z. B. ET200S HS für IRT Hohe Performance)
  - Betrieb eines SINAMICS S120 als IO-Device
  - Datenaustausch zwischen Controllern über IRT Hohe Performance (z. B. Verteilter Gleichlauf)
- RT - Realtime Ethernet
  - Betrieb von RT - Peripherie (z. B. ET 200S, ET 200pro)
  - ASi-Link über IE/AS-Interface Link PN IO für den Netzübergang PROFINET IO auf AS-Interface
  - SINAMICS als PROFINET IO mit RT Gerät
- TCP/IP, UDP, HTTP, … Standard Ethernet Dienste

Hinweis

Bei Mischbetrieb von IRT Hohe Performance und RT ist darauf zu achten, dass die IRT Hohe Performance-fähigen Geräte direkt miteinander verbunden sind. D. h., es darf kein **nicht** IRT-fähiges Gerät zwischen die IRT-Geräte geschaltet werden.

Hinweis

Mit SIMOTION SCOUT können Sie maximal zu 10 PROFINET Teilnehmern gleichzeitig ONLINE gehen. Wenn Sie SIMATIC NET installiert haben, ist es möglich, mehr als 10 Verbindungen aufzubauen.

PROFINET V2.2

SIMOTION SCOUT unterstützt PROFINET V2.2. Ältere Versionen werden standardmäßig in der Hardware nicht mehr unterstützt bzw. müssen umgerüstet werden.

Beim Einfügen von der SIMOTION Geräten bzw. SINAMICS Antrieben werden im SIMOTION SCOUT nur PROFINET V2.2 Versionen eingefügt. Wenn Sie ältere Versionen projektieren wollen, müssen Sie die Hardware explizit in HW Konfig als PROFINET V2.1 einfügen.

Hinweis

Alle IRT-Teilnehmer müssen der Norm PROFINET V2.2 entsprechen. Mischkonfigurationen mit älteren Versionen sind nicht zulässig.

Standardmäßig wird die Hardware als PROFINET V2.2 ausgeliefert. Sollten Sie bestehende Anlagen hochrüsten bzw. neue Hardware rückrüsten, kontaktieren Sie bitte den Siemens Support.

Übersicht der Projektierung bei PROFINET V2.2 mit STEP7 5.4 SP4 und SCOUT V4.1.2

Tabelle 4- 3 PROFINET Projektierung

Projektierungsschritt	IRT nach PN V2.2
Konfiguration der RT-Klasse in HW Konfig	IRT Hohe Performance
Konfiguration von SINAMICS S120 über GSD	Ab GSD V2.2
Konfiguration von SINAMICS S120 über DeviceOM	Wird unterstützt
SIMOTION	Ab 4.1.2
SINAMICS	Ab 2.5.1.10
SCALANCE X200 IRT Switch	Ab V4.1

4.2.2 Taktuntersetzung

4.2.2.1 Taktuntersetzung mit PROFINET IO an SIMOTION Geräten

Beschreibung (PROFINET IO mit IRT Hohe Performance)

Eine taktsynchrone Applikation (z. B. Lageregler) auf einem IO-Controller synchronisiert sich auf den Sendetakt von IRT Hohe Performance. Sie kann sich aber auch auf ein Vielfaches des Sendetaktes der Daten synchronisieren. Dieses Vielfache wird als CACF (Controller Application Cycle Factor) bezeichnet. Die Taktuntersetzung wird im SCOUT beim Ablaufsystem über das Kontextmenü **Experte > Systemtakte einstellen** projektiert.

Beispiel: Die Daten auf dem Netz werden mit einem Sendetakt von 1 ms übertragen. Der Servo soll aber mit 2 ms laufen. Deshalb muss der Applikationszyklus 2 sein. Dazu stellen Sie bei **Systemtakte einstellen** am Servo das Verhältnis 2 ein.

Hinweis

Taktsynchronität ist mit PROFINET IO mit IRT Hohe Flexibilität nicht möglich.

Die Einstellung des CACF erfolgt am IO-Device, siehe z. B. SINAMICS S120 einfügen und projektieren (Seite 102).

Ab V4.2 zwei Servo-Takte (Servo, Servo_fast)

Ab V4.2 wird für schnelle Applikationen der Servo_fast als 2. Servo Task eingeführt. Mit dem zweiten Servo-Takt können Sie zwei Bussysteme in unterschiedlichen Applikationstakten betreiben (PROFINET und PROFIBUS). Für beide Applikationstakte steht je ein zugeordneter Servo und IPO-Takt zur Verfügung. Damit können Sie Ihre Applikation in einen langsamen (Servo und IPO) und einen schnellen Teil (Servo_fast und IPO_fast) aufteilen. Ist ein Servo_fast konfiguriert, so ist der Servo_fast im Verhältnis 1:1 an den Sendetakt von PROFINET IRT gekoppelt. Der Takt des Servo kann dann wieder als Vielfaches des Servo_fast eingestellt werden. Der CACF ist dann weiterhin der Faktor zwischen Sendetakt bzw. Servo_fast und Servo.

Die im folgendem beschriebenen Funktionen beziehen sich im Wesentlichen auf Konfigurationen ohne die Option 2. Servo (Servo_fast)

Hinweis

Die detaillierte Beschreibung zur Option Servo_fast finden Sie im *Handbuch SIMOTION SCOUT Basisfunktionen* Kapitel 6.

Beschreibung

Eine Untersetzung zum Sendetakt mit SIMOTION Controllern ist mit PROFINET mit IRT Hohe Performance unter folgenden Bedingungen möglich:

- Der SINAMICS Integrated einer D4xx und eine taktsynchrone DP-Master-Schnittstelle laufen immer gleich zum Servo-Takt.
- Bei einer SIMOTION P350 läuft die taktsynchrone DP-Master-Schnittstelle immer gleich zum Servo-Takt.
- Bei einem Antrieb (z. B. S120), der über den SINAMICS Integrated oder als externes Antriebsgerät angeschlossen ist, muss der Servo-Takt **immer** im ersten Sendetakt gerechnet sein, d. h., es ist zwar eine Untersetzung möglich, jedoch muss der Lageregler innerhalb eines Sendetaktes gerechnet werden.

Folgende allgemeine Bedingungen gelten für Takte und Taktuntersetzungen für einen SIMOTION-Controller

- Wenn IRT Hohe Performance für ein SIMOTION-Gerät projektiert ist, aber das SIMOTION-Gerät selbst keine IRT-Daten sendet bzw. empfängt (z. B. nur Durchleiter für IRT-Daten), ist der Servo-Takt nicht auf den Sendetakt synchronisiert. Nur die PROFINET-Schnittstelle ist auf den Sendetakt synchronisiert, z. B.
  - Nur TCP/IP über PROFINET-Schnittstelle
  - Nur RT-Devices an der PROFINET-Schnittstelle
  - PROFINET-Schnittstelle nur *Durchleiter* für IRT Hohe Performance Daten zu anderen Geräten

Hinweis

Nur bei dem SIMOTION-Gerät, das Sync-Master ist, synchronisiert sich das Gerät automatisch auf den übergeordneten Bus auf. Ist keine PROFIBUS DP-Schnittstelle des SIMOTION Geräts als taktsynchroner DP-Master projektiert, synchronisieren sich auch die Sync-Slaves automatisch auf.

Ist eine PROFIBUS DP-Schnittstelle als taktsynchroner DP-Master für unterlagerte Antriebe konfiguriert, muss das Gerät per Applikation aufsynchronisiert werden. Dies wird in der StartupTask über den Befehl _enabelDPInterfaceSynchronizationMode programmiert.

Kombinationen von Takten und Taktquellen bei PROFIBUS und PROFINET IO

- Servo kann in ganzzahligen Vielfachen (1, 2, ... n) zum Sendetakt untersetzt werden.
- Ist ein Servo_fast konfiguriert, so ist er fest im Verhältnis 1:1 zum Sendetakt eingestellt, CACF wie bisher als Verhältnis Sendetakt/Servo_fast zu Servo einstellbar.
- Takte des SINAMICS Integrated und taktsynchrone DP-Masterschnittstellen müssen gleich zum Servo-Takt laufen.

4.2.2.2 Taktuntersetzung bei Peripheriezugriffen

Beschreibung für Datenübertragung PROFINET IRT

Bei Taktuntersetzung (PROFINET und PROFIBUS) ist folgendes zu beachten:

- PROFINET IO IRT Daten werden immer zu Begin des Servo-Taktes gelesen und am Ende des Servo-Taktes geschrieben.
- PROFINET IO RT Daten werden zu Begin des IPO- oder des IPO2-Taktes gelesen und am Ende des IPO-Taktes geschrieben.

- Am Ende einer IPOSynchronousTask wird das Prozessabbild mit dem nächstmöglichen Servo (Data Out) ausgegeben (= reaktionszeitoptimiert). Dieses kann bei Untersetzung Servo-Takt zu IPO-Takt dazu führen (Servo < IPO), dass die Daten einen /mehrere Servo-Takte früher oder später innerhalb eines IPO-Taktes ausgegeben werden, wenn die Peripheriezugriffe über die IPOSynchronousTask erfolgen. Dies ist der Fall, wenn die Laufzeit des IPO-Taktes nicht konstant ist und daher bei schnellerem Servo-Takt die Daten früher oder später am Bus übertragen werden.
- Am Ende der Servo-Ablaufebene wird das Prozessabbild der ServoSynchronousTask mit dem nächstmöglichen Bus-Takt ausgegeben (= reaktionszeitoptimiert).
- Auch wenn PROFINET und Untersetzung PROFINET-Takt zu Servo-Takt (PROFINET < Servo) werden die Daten immer im ersten PROFINET-Takt ausgegeben.
- Bei PROFIBUS werden die Daten immer mit dem ersten Bus-Takt ausgegeben, da die Servo-Ablaufebene immer mit dem ersten Bus-Takt abgeschlossen sein muss. Bei unterschiedlicher Laufzeit der Servo-Ablaufebene in den einzelnen Takten kann somit die Klemme-Klemme-Zeit variieren.

Soll statt eines reaktionszeitoptimierten Verhaltens eine immer konstante Reaktionszeit erreicht werden, so muss:

- Bei PROFIBUS:
  - ein Untersetzungsverhältnis Servo : IPO = 1 : 1 eingestellt werden, damit Peripheriezugriffe aus der IPOSynchronousTask immer taktsynchron erfolgen.
  - Anmerkung: Peripheriezugriffe aus der ServoSynchronousTask sind bei PROFIBUS immer taktsynchron
- Bei PROFINET:
  - ein Untersetzungsverhältnis Bus-Takt : Servo : IPO = 1 : 1 : 1 eingestellt werden, damit Peripheriezugriffe aus der IPOSynchronousTask immer taktsynchron erfolgen
  - ein Untersetzungsverhältnis Bus-Takt : Servo = 1 : 1 eingestellt werden, damit Peripheriezugriffe aus der ServoSynchronousTask immer taktsynchron erfolgen

4.2.2.3 Einstellbare Bustakte bei Taktuntersetzung an SIMOTION-Geräten

Übersicht über die möglichen Bustakte

	PROFIBUS	PROFINET IRT Hohe Performance	PROFINET IRT Hohe Performance	Servo Servo_fast
	Minimal	Minimal	Maximal	Minimal
SINAMICS S120 CU320	1 ms	0,5 ms	4,0 ms	0,5 ms
SINAMICS S120 CU310	1 ms	0,5 ms	4,0 ms	0,5 ms
SINAMICS S120 CU310-2	1 ms	0,5 ms	4,0 ms	0,5 ms

	PROFIBUS	PROFINET IRT Hohe Performance	PROFINET IRT Hohe Performance	Servo Servo_fast
	Minimal	Minimal	Maximal	Minimal
SINAMICS S120 CU320-2	1 ms	0,5 ms	4,0 ms	0,5 ms
SINAMICS S110 CU305	1 ms	1 ms	4 ms	1 ms
C230-2	1,5 ms	-	-	1,5 ms
C240 PN	1 ms	0,5 ms	4,0 ms	0,5 ms
C240	1 ms	-	-	0,5 ms
D410 PN	-	0,5 ms	4,0 ms	2,0 ms
D410 DP	2 ms	-	-	2,0 ms
D425	2 ms	0,5 ms	4,0 ms	2,0 ms
D435	1 ms	0,5 ms	4,0 ms	1,0 ms
D445/D445-1	1 ms	0,5 ms	4,0 ms	0,5 ms
D445-2 DP/PN	1 ms	0,25 ms[1)]	4,0 ms	0,25 ms [1)]
D455-2 DP/PN	1 ms	0,25 ms[1)]	4,0 ms	0,25 ms [1)]
P350-3	1 ms	0,25 ms	4,0 ms	0,25 ms
P320-3	-	0,25 ms	4,0 ms	0,25 ms
ET200S HS	-	0,25 ms	4,0 ms	0,25 ms

1) Erläuterung:

- 0,5 ms in Verbindung mit SINAMICS S120 (inkl. SINAMICS Integrated / CX32-2)
- 0,25 ms in Verbindung mit SERVO_fast und IPO_fast für eine schnelle I/O-Verarbeitung bzw. hochperformante Hydraulikanwendungen. Die Sensorik und Aktorik wird hierbei über High Speed PROFINET IO Peripheriebaugruppen angeschlossen

Taktuntersetzung mit PROFINET IO

Takt	Servo		IPO		IPO2	
	Min	Max	Min	Max	Min	Max
Takt	1 x Bus	16 x Bus	1 x Servo	6 x Servo	2 x IPO	64 x IPO

Takt	Servo_fast		IPO_fast	
	Min	Max	Min	Max
Takt	1 x Bus	1 x Bus	1 x Servo_fast	4 x Servo_fast

4.2.3 Tasksystem und Zeitverhalten

4.2.3.1 Übersicht SIMOTION Tasksystem und Systemtakte

Übersicht

Werden über PROFINET IO IRT-Daten über den Bus übertragen, sind die Takt-Laufzeiten zwischen Lesen und Schreiben der Daten, z. B. Achsdaten, abhängig davon, in welcher Task im Ablaufsystem die Applikation ausgeführt wird. Beispiele für Applikationen in unterschiedlichen Tasks (Ablaufebenen) finden Sie in den folgenden Kapiteln.

4.2.3.2 Background-, Motion- und IPOsynchronous Task

Motion-/BackgroundTask

Die Daten werden über PROFINET IO mit IRT Hohe Performance über den Bus übertragen und vom Kommunikationsinterface zu Beginn des Servos übernommen. Die Auswertung der Logiksignale erfolgt typisch in einer Motion oder der Background Task. Dort wird entschieden, welche Maschinenfunktion aktiviert wird, z. B. "verfahre Achse lagegeregelt". Das notwendige Verfahrprofil wird im nächsten IPO Takt gerechnet. Aus den dort ermittelten Lagesollwerten werden im nächsten Servo-Takt die Drehzahlsollwerte für die Ansteuerung der Achse errechnet. Diese werden im nächsten Zyklus über PROFINET IO mit IRT Hohe Performance an den Antrieb übertragen.

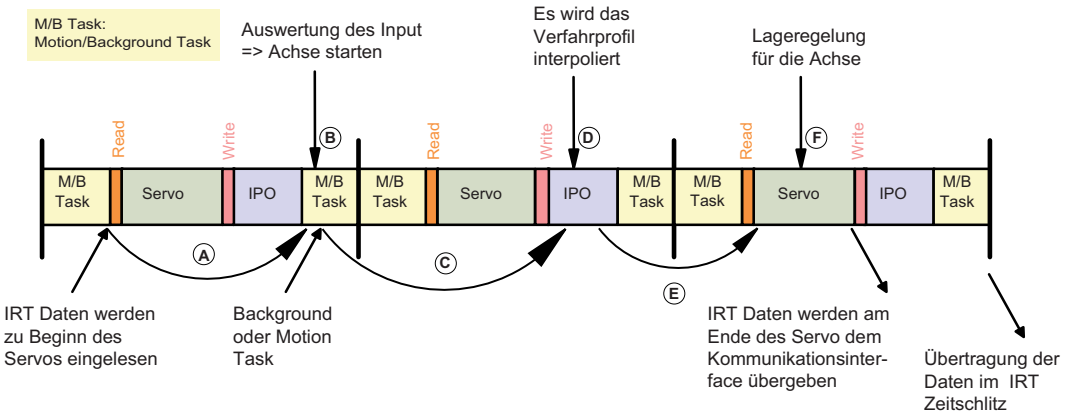


Bild 4-8 Logikauswertung für eine Achse in der Background- bzw. MotionTask

Randbedingungen

Die Option 2 Servo_Takte ist nicht aktiviert, d.h. kein Servo_fast und IPO_fast. Bustakt, Servo-Takt und IPO-Takt sind im Verhältnis 1:1:1. Bei anderen Verhältnissen können sich längere Reaktionszeiten ergeben. Bei 1:1:2 kann sich die Abarbeitung des IPOs auf zwei Servo-Takte ausdehnen, damit kann sich die Reaktionszeit um einen Servo-Takt erhöhen.

Weiterhin kann sich die Bearbeitung der BackgroundTask über mehrere Servo-Takte erstrecken. Damit kann vom System nicht sichergestellt werden, dass im ersten Servo-Takt die Daten ausgewertet werden. Dadurch kann sich ebenfalls die Reaktionszeit erhöhen.

Auch die Zuordnung der Variablen zu einem Prozessabbild hat Auswirkung auf die Reaktionszeit. Die Prozessabbilder werden nicht nach Aktualisierung der Variablen, sondern am Ende der jeweiligen Task den anderen Tasks bzw. dem Kommunikationsinterface zur Verfügung gestellt.

IPOsynchronousTask

Um das Zeitverhalten zu optimieren und das synchrone Auslösen von Aktionen zu ermöglichen, z. B. gleichzeitiges Starten von Achsen, besteht die Möglichkeit, den Teil der Applikation, der Achsbefehle auslöst, in der IPOsynchronousTask zu bearbeiten. Diese wird vor dem IPO gerechnet. Dadurch kann vor der Abarbeitung des IPOs der Achsbefehl abgesetzt und im IPO dann der daraus resultierende Lagesollwert ermittelt werden. Aus diesem wird dann im nächsten Servo-Takt der Drehzahlsollwert für den Antrieb errechnet. Am Ende des Servos werden die Daten ans Kommunikationsinterface übergeben und im nächsten PROFINET IRT-Sendetakt übertragen. Im Gegensatz zur Bearbeitung in MotionTask/BackgroundTask mit einer Reaktionszeit entsprechend der maximalen BackgroundTask-Laufzeit + einen IPO-Takt + einen Servo-Takt ergibt sich ein sicheres Reaktionsverhalten mit einem IPO-Takt + einen Servo-Takt bis zur neuen Datenausgabe.

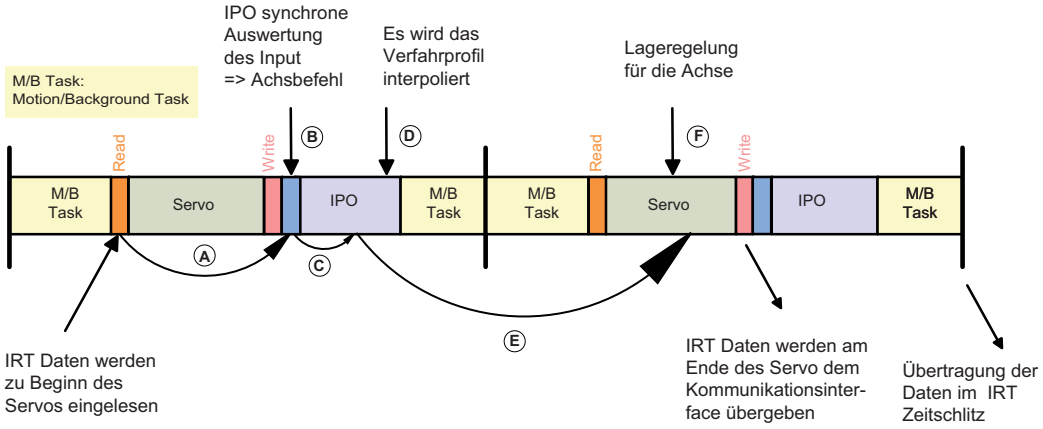


Bild 4-9 Logikauswertung für eine Achse in der IPOsynchronousTask

4.2.3.3 ServoSynchronousTask

ServoSynchronousTask

Es besteht die Möglichkeit das Zeitverhalten weiter zu optimieren und die Reaktionszeit auf einen Servo-Takt zu reduzieren. Dies kann für schnelle Istwertgleichläufe, z. B. für das fliegende Messer/Schere genutzt werden. Dabei wird der Teil der Applikation, der die Achsbefehle für ausgewählte Achsen auslöst, in der ServoSynchronous Task bearbeitet. Weiterhin wird der IPO-Systemanteil für die betroffenen Achsen in der Servo-Task vor dem Lageregler gerechnet. Dadurch können die Drehzahlsollwerte bereits im nächsten IRT-Zeitschlitz übertragen werden.

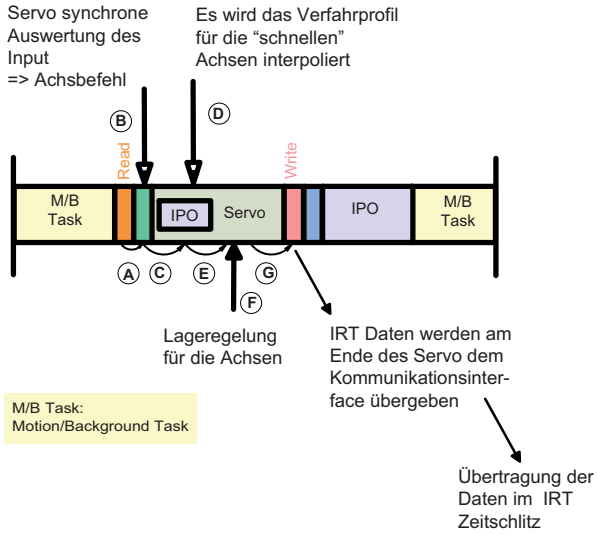


Bild 4-10 Logikauswertung für eine Achse in der ServoSynchronousTask

Randbedingungen

Die Verwendung dieser Funktion erhöht die CPU-Belastung und damit den Servo-Takt. Daher sollte die Funktion nur bei Bedarf genutzt werden.

Aktivierung

Dieses Feature muss explizit für die Achsen im SIMOTION SCOUT unter der Konfiguration der Achse aktiviert werden.

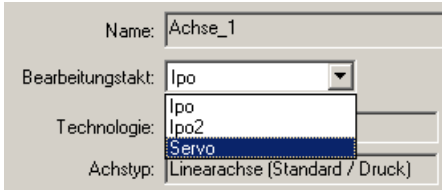


Bild 4-11 Bearbeitungstakt der Achse festlegen

Ist die Option 2 Servo-Takte mit Servo_fast konfiguriert, so kann die Achse auch dem Bearbeitungstakt Servo_fast zugeordnet werden, wenn diese am schnelleren Bus hängt.

Lageregelung

Die Reaktion des Lagereglers erfolgt innerhalb eines Servo-Taktes. Die Daten werden zu Beginn des Servos aus dem Kommunikationsinterface ausgelesen. Im Servo wird der Lageregler gerechnet und am Ende des Servos die neuen Drehzahlsollwerte in das Kommunikationsinterface kopiert und damit im nächsten IRT-Zeitschlitz übertragen.

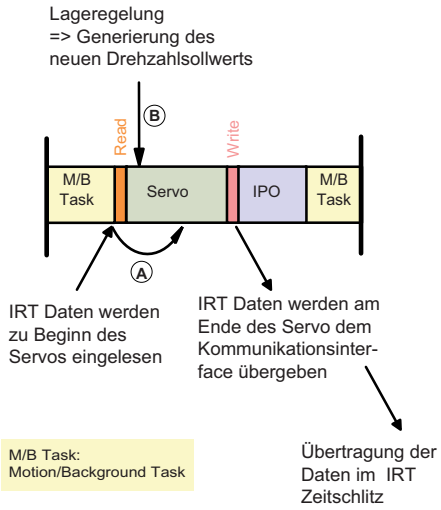


Bild 4-12 Zeitverhalten Lageregelung bei ServoSynchronousTask

4.2.3.4 Schnelle I/O-Verarbeitung in der ServoSynchronousTask

Schnelle I/O-Verarbeitung in ServoSynchronousTask

Die Auswertung von schnellen I/O, z B. ET200S HS, erfolgt in der Servo synchronen Task. Damit ergibt sich eine Reaktionszeit von einem Takt im System. Bezogen auf das Klemme-Klemme-Reaktionsverhalten ergibt sich eine Verzögerung von Ti + Servo-Takt + To.

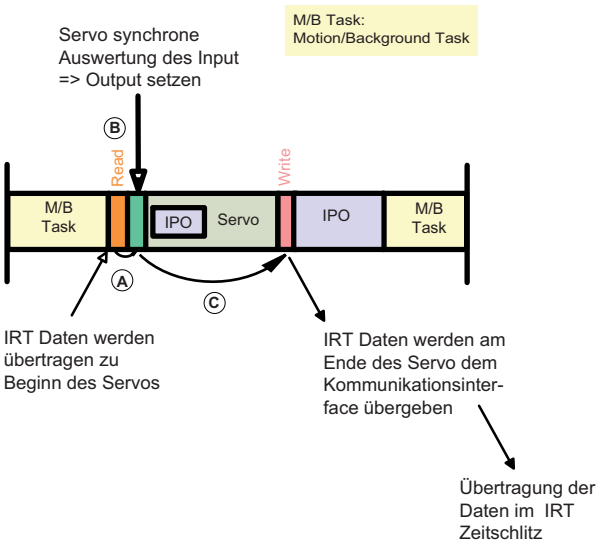


Bild 4-13 Fast IO in der ServoSynchronousTask

Ist die Option 2 Servo-Takte mit Servo_fast konfiguriert, so kann die schnelle IO-Bearbeitung auch im Takt Servo_fast erfolgen in der ServoFastSynchronousTask. Voraussetztung ist, dass die IO-Peripherie am schnelleren Bus konfiguriert ist.

4.2.4 Zusammenhang Sync-Domain und IO-Systeme

Die Geräte mehrerer IO-Systeme können von einem einzigen Sync-Master synchronisiert werden, sofern sie am selben Ethernet-Subnetz angeschlossen sind und einer Sync-Domain angehören.

Umgekehrt gilt, dass ein IO-System nur einer einzigen Sync-Domain angehören kann.

4.2.5 Redundanter Sync-Master

Beschreibung

Für bestimmte Anlagen, z. B. Druckmaschinen, ist es erforderlich, dass Anlagenteile standalone oder in Verbindung mit einem weiteren synchron betrieben werden können. Wenn die Gesamtanlage nur einen Sync.-Master hat, wäre der jeweils andere Teil nicht für sich alleine funktionsfähig. Daher wurde die Funktion "Redundanter Sync-Master" realisiert. Dabei wird für jede Teilmaschine ein Sync-Master definiert. Einer von beiden wird als "Sync-Master" der andere als "Sync-Master (redundant)" definiert. Solange die Anlage kombiniert genutzt wird, synchronisiert sich der "Sync- Master (redundant)" auf den Sync-Master auf.

Entfällt der Sync-Master, läuft der Sync-Master (redundant) autark weiter und synchronisiert die ihm zugeordneten Sync-Slaves. Die dem Sync-Master zugeordneten Sync-Slaves verlieren die Synchronisation und laufen nicht weiter.

Beschränkungen

Die beiden Sync-Master müssen direkt mit einem Kabel ohne Switch verbunden sein. Fällt die Übertragungsstrecke zwischen dem Sync-Master und Sync-Master (redundant) aus, sodass 2 Teilnetze mit je einem Sync-Master entstehen, bleiben die beiden Teilnetze auf den jeweils verbliebenen Sync-Master synchronisiert. Es entstehen damit zwei unabhängig voneinander synchronisierte Teilnetze, die u. a. durch eine Temperaturdrift der Quarze auseinanderlaufen. Nach der Wiederherstellung der Übertragungsstrecke kann sich der Sync-Master (redundant) nicht stoßfrei auf den Sync-Master aufsynchronisieren, d. h. die dem Sync-Master (redundant) zugeordneten Antriebe würden für kurze Zeit die Synchronisation verlieren und ausfallen. Für eine Synchronisation muss die Applikation gestoppt werden und neu aufsynchronisiert werden.

Zweiten Sync-Master projektieren

1. Fügen Sie eine zweite SIMOTION-Baugruppe ein und projektieren Sie PROFINET entsprechend Ihren Vorgaben.
2. Klicken Sie mit der rechten Maustaste auf das PROFINET-Board um den Dialog **Eigenschaften - <PROFINET-Board> -- (R0/S2.6)** aufzurufen.
3. Wählen Sie im Register **Synchronisation** unter **Synchronisationsart** den Eintrag **Sync-Master (redundant)** aus.

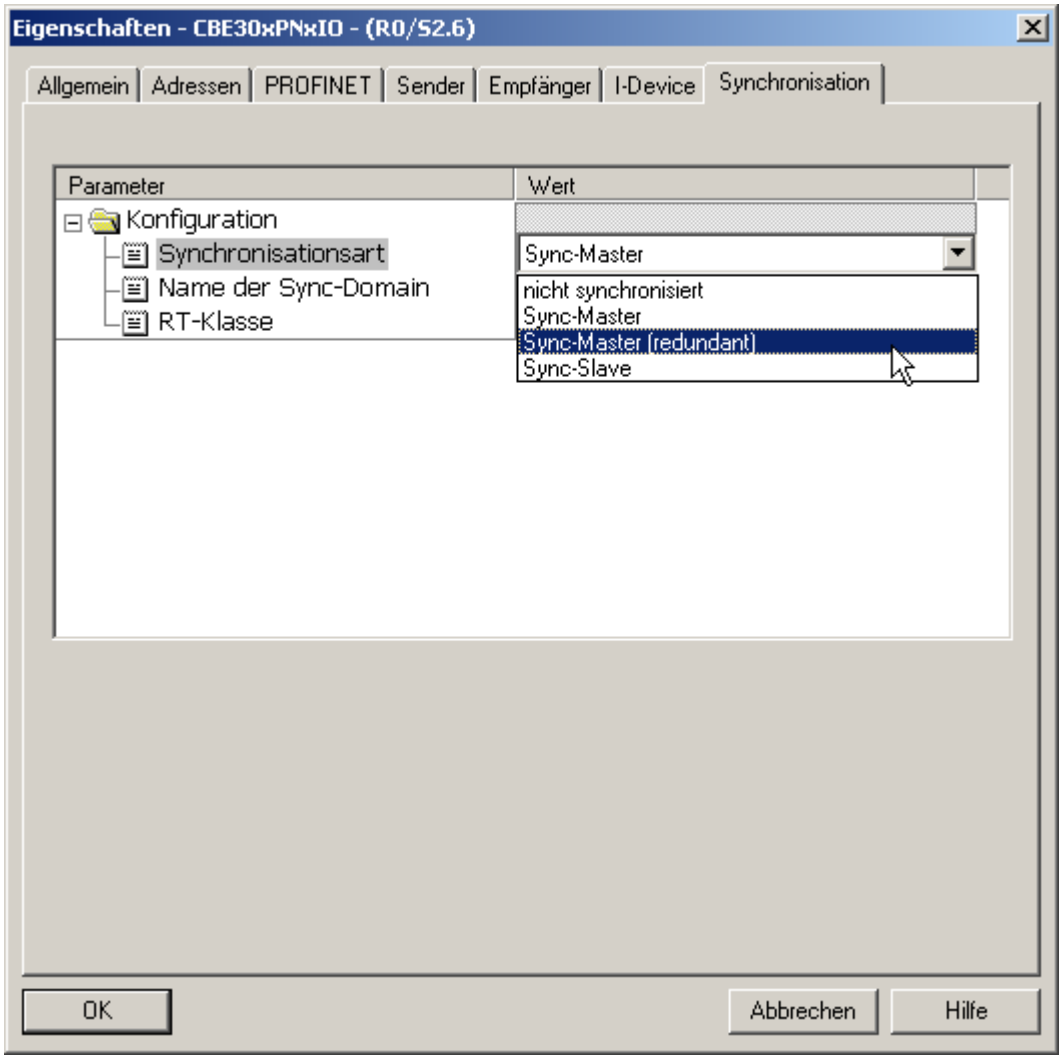


Bild 4-14 Zweiten Sync-Master projektieren

4.2.6 Mengengerüste

Für IO-Controller der SIMOTION Plattform gelten folgende Maximalwerte.

Es sind maximal 64 Kommunikationsbeziehungen möglich, diese können sich wie folgt aufteilen:

- Anschluss von maximal 64 IO-Devices.
- Maximal 64 RT.
- Maximal 64 IRT Hohe Performance Devices.
- Maximal dürfen bis zu 64 Controller-Controller-Querverkehrsbeziehungen zwischen IO-Controllern aufgebaut werden.
- Wenn eine SIMOTION als I-Device projektiert wurde, zählt diese als Device mit, d.h. es sind dann nur noch 63 Devices anschließbar.
- Eine SIMOTION kann Daten von bis zu 64 anderen SIMOTION bei Controller-Controller QVK empfangen, kann aber an beliebig viele SIMOTION Daten senden.
- Beim Controller-Controller-QVK ist noch die Datenmenge zu berücksichtigen. Ein Querverkehr wird nur bis zu einer bestimmten Datenmenge als eine Verbindung gezählt. Wenn ein zweites Telegramm benötigt wird, entfallen auf den Querverkehr zwei Verbindungen.

Mischbetrieb von IO-Devices und Controller-Controller Querverkehr

Die mögliche Anzahl von Geräten im Mischbetrieb können Sie sich über folgende Formeln berechnen:

IRT Hohe Performance

```
RT + IRT Hohe Performance-IO-Device + Querverkehrsframes <= 64
```

Hinweis

Bei einer Querverkehrsbeziehung ist nicht die Anzahl der projektierten Slots (Zeilen in den Laschen Empfänger) (siehe Empfänger projektieren (Seite 117)) für die IRT Hohe Performance Querverkehrsprojektierung gemeint, sondern die Anzahl der Ethernet-Frames, die für den Querverkehr empfangen werden. Ein Ethernetframe kann max. 768 Byte Querverkehrsnutzdaten enthalten.

Ein Slot hat maximal 254 Bytes und im Querverkehr können 3072 Bytes Nutzdaten (aufgeteilt auf 4 Frames je 768 Byte) ausgetauscht werden. Der Wert ist abhängig von den gewählten Slotgrößen. Es können daher mehrere Slots von einem Sender empfangen werden, die aber in einem Telegramm übertragen werden müssen.

Jeder Provider sendet seine Querverkehrs-Daten in einem Ethernet-Frame. Jede andere SIMOTION kann diese Daten in diesem Frame lesen. So haben wir eine zählende Verbindung zu jedem sendenden SIMOTION.

Beim Controller-Device-Verkehr hat ein Frame die Nutzdatengröße von 1440 Bytes.

HW Konfig prüft gemäß den oben genannten Formeln das projektierte Mengengerüst beim Übersetzen des Projektes.

Adressraum

Im logischen Adressraum eines IO-Controllers dürfen jeweils für Input- und für Output-Daten maximal 4 Kilobyte für PROFINET IO-Daten belegt werden. Der Rest des insgesamt 16 Kilobyte großen Adressraumes kann z. B. durch PROFIBUS-Daten oder Diagnosedaten belegt werden.

4.3 PROFINET IO mit SIMOTION projektieren

4.3.1 Neues ab SIMOTION SCOUT V4.2

Synchronisationsverfahren IRT

SIMOTION nur IRT Hohe Performance. Dieses Synchronisationsverfahren wird automatisch bei PROFINET IRT eingestellt.

Controller mit integrierter PN-Schnittstelle

Mit SIMOTION D4x5-2 DP/PN sind neue SIMOTION D Controller mit integrierten PN-Schnittstellen verfügbar.

Taktsynchronisation für Applikationen wurde vereinfacht

Die typische taktsynchronen Applikationen für Motion Control wurde stark vereinfacht. Die Berechnung der Ti/To Zeitkonstanten kann für alle IO-Devices automatisch erfolgen. Die Projektierung in HW Konfig wurde dahingehend erweitert, dass am Controller die Objekteigenschaften der Taktsynchronen Tasks auf automatisch gesetzt werden können.

I-Device

Mit Step 7 V5.5 wurde die I-Device Funktionalität verbessert. Bei SIMOTION SCOUT Projekten < V4.2 müssen beim Hochrüsten auf V4.2 einige Punkte beachtet werden. Mehr dazu im Kapitel I-Device.

I-Device-F-Proxy

Mithilfe des I-Device-F-Proxys können Sie eine PROFIsafe-Projektierung mit F-Host (F-CPU SIMATIC) an PROFINET mit SIMOTION Geräten (SIMOTION D4xx, SIMOTION P3xx, SIMOTION C240 PN) für die unterlagerten Antriebe erstellen.

Zweiter Servo-Takt (Servo_fast)

Mit dem zweiten Servo-Takt können Sie zwei Bussysteme in unterschiedlichen Applikationstakten betreiben. Für beide Applikationstakte steht je ein zugeordneter Servo und IPO-Takt zur Verfügung. Damit können Sie Ihre Applikation in einen langsamen und einen schnellen Teil (Servo_fast und IPO_fast) aufteilen.

4.3.2 Vorgehensweise zur Projektierung von PROFINET IO mit IRT Hohe Performance

Vorgehensweise

Für die Projektierung von PROFINET IO mit IRT Hohe Performance V2.2 müssen Sie Folgendes durchführen:

1. Einfügen der SIMOTION-Baugruppe.
   Sie wählen zuerst das **Gerät** und dann die **Geräteausprägung**. Es sind nur PROFINET V2.2 Varianten möglich. Bei mit PN gekennzeichneten Geräten ist schon eine PROFINET-Schnittstelle integriert.
   - Bei SIMOTION C und SIMOTION P wählen Sie nur noch die SIMOTION Version z. B. V4.2.
   - Bei SIMOTION D wählen Sie abhängig von der Ausprägung die SIMOTION Version V4.2, die Optionsbaugruppe und den SINAMICS Antrieb und Version.
     Die Optionsbaugruppe CBE30 für PROFINET können Sie hier auswählen oder fügen diese später in HW Konfig aus dem Hardwarekatalog unter **SIMOTION Drive Based > SIMOTION D4xx > 6AUxx > V4.2 - PN-V2.2xx** ein.
2. IO-Devices einfügen:
   Fügen Sie IO-Devices aus dem Hardware-Katalog von HW Konfig in das I/O-System ein. Die IO-Devices finden Sie im Hardwarekatalog unter **PROFINET IO**.
3. Sync-Domain projektieren und Sendetakt festlegen:
   Projektieren Sie den PROFINET IO Teilnehmer als Sync-Master (Taktgeber) und definieren Sie die dazugehörigen Sync-Slaves. Legen Sie den Sendetakt fest.
4. Topologie erstellen:
   Definieren Sie die Topologie, d. h., legen Sie fest, wie die einzelnen Ports der PROFINET IO Geräte untereinander verschaltet sind. Die Projektierung der Topologie ist nur für PROFINET IO mit IRT Hohe Performance notwendig. Wenn die topologiebasierte Taufe genutzt werden soll, ist die Topologieprojektierung für alle PROFINET-Geräte erforderlich
5. Controller - Controller Querverkehr:
   Legen Sie fest, welche Adressbereiche zum Senden und welche zum Empfangen verwendet werden sollen.

4.3.3 SIMOTION D einfügen und projektieren

4.3.3.1 Allgemeines SIMOTION D einfügen und projektieren

Allgemeines

Sie haben ein Projekt angelegt und wollen eine SIMOTION D unter der Verwendung einer PROFINET-Schnittstelle projektieren.

Bei SIMOTION D verfügen die Control Units abhängig von der Ausprägung über eine Onboard PROFINET-Schnittstelle (D410 PN, D4x5-2 DP/PN) oder über eine Optionsbaugruppe CBE30 für PROFINET (D4x5).

Die CBE30 können Sie im Dialog auswählen oder fügen diese später über HW Konfig aus dem Hardwarekatalog unter SIMOTION Drive Based > SIMOTION D4x5 > 6AUxx > V4.2 - PN-V2.2xx ein.

Realisierung PROFINET-Schnittstelle:

- bei D410 PN und D4x5-2 DP/PN bereits integriert
- bei D4x5 optional verfügbar (Optionsbaugruppe CBE30)

Das Vorgehen wird in den nachfolgenden Kapiteln beschrieben.

4.3.3.2 SIMOTION D4x5-2/D410 PN einfügen und projektieren

Allgemeines

Sie haben ein Projekt angelegt und wollen eine SIMOTION D4x5-2 DP/PN bzw. SIMOTION D410 PN unter der Verwendung der internen PROFINET-Schnittstelle projektieren. Am Beispiel einer SIMOTION D4x5-2 DP/PN wird die Vorgehensweise dargestellt.

Vorgehensweise

1. Klicken Sie auf **SIMOTION Gerät einfügen** im Projektnavigator, damit der Geräte-Auswahldialog geöffnet wird.
2. Wählen Sie unter **Gerät** SIMOTION D und klicken Sie auf eine **Geräteausprägung** z. B. D455-2 DP/PN.

3. Wählen Sie die **SIMOTION Version** V4.2 und **SINAMICS S120 Integrated.**

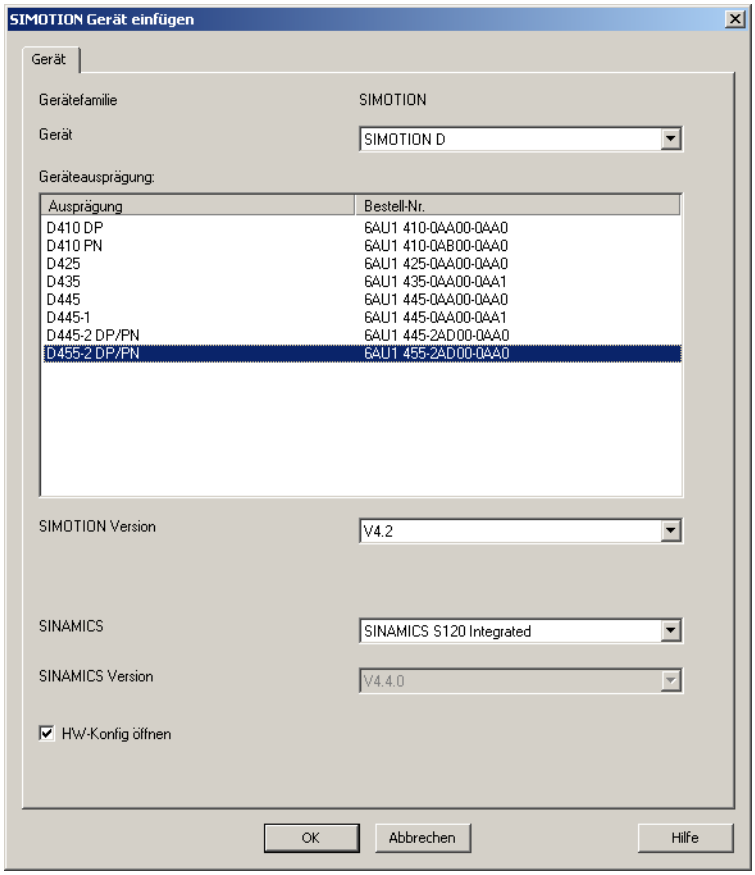


Bild 4-15 SIMOTION Gerät einfügen

4. Aktivieren Sie die Checkbox **HW Konfig öffnen**, um z. B. eine Optionsbaugruppe oder ein IO-System einzufügen.
5. Bestätigen Sie mit **OK**.

6. Der Dialog zum Anlegen eines PROFINET-Subnetzes wird eingeblendet. Erstellen Sie mit **Neu...** ein neues Subnetz und tragen Sie die **IP-Adresse** und die **Subnetzmaske**. Bestätigen Sie mit **OK**.

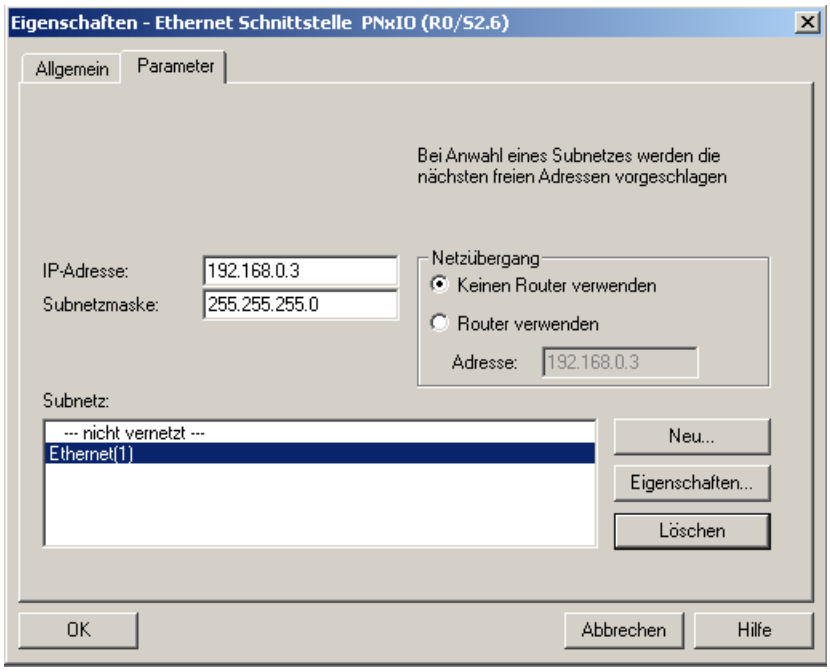


Bild 4-16 Neues Ethernet-Subnetz anlegen

7. Wählen Sie im folgenden Dialog die PG/PC-Schnittstelle aus und bestätigen Sie mit OK. Das Gerät wird eingefügt, HW Konfig wird geöffnet und es wird die Baugruppe mit dem projektierten PROFINET-Subnetz angezeigt.

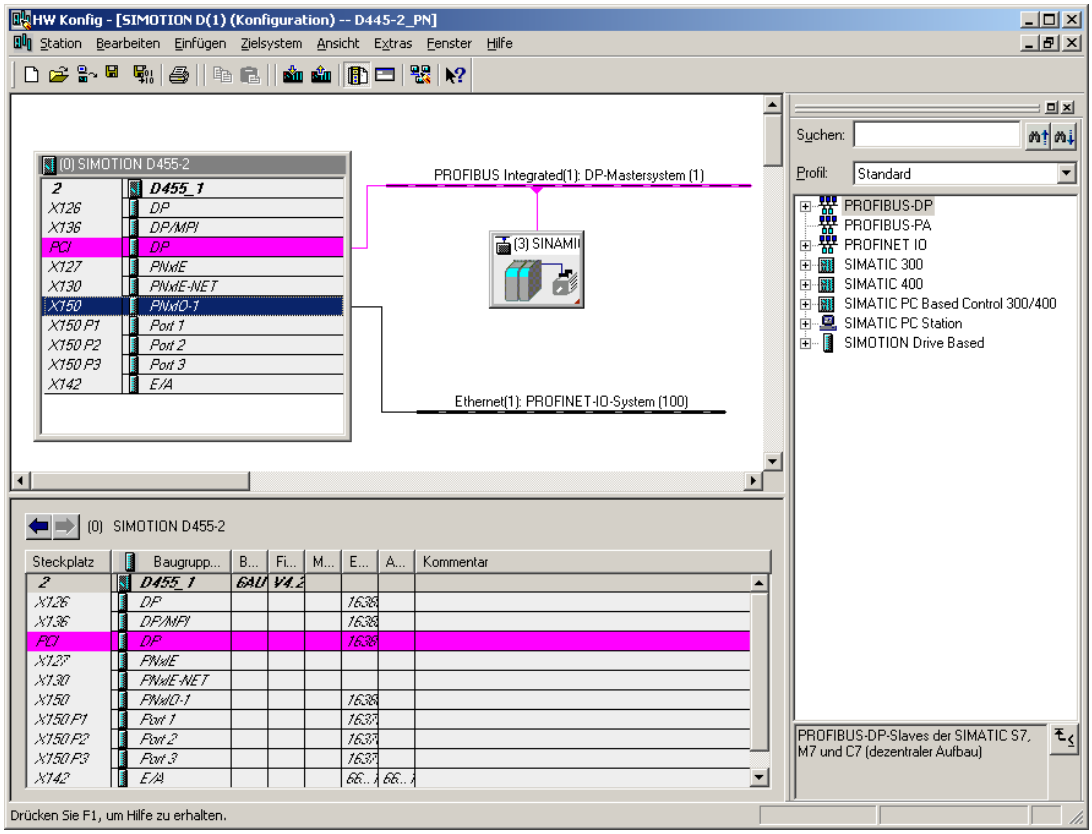


Bild 4-17 SIMOTION D455-2 DP/PN in HW Konfig

Siehe auch

CBE30-PROFINET-Board einfügen und projektieren (Seite 82)

4.3.3.3 SIMOTION D4x5 inkl. CBE30 einfügen und projektieren

Voraussetzung

Sie haben bereits ein Projekt angelegt und möchten jetzt eine SIMOTION D4x5 mit der Optionsbaugruppe CBE30 für PROFINET einfügen. Bei SIMOTION D4x5 ohne integrierter PN-Schnittstellen müssen Sie die Optionsbaugruppe CBE30 verwenden.

Vorgehensweise:

1. Klicken Sie auf **SIMOTION Gerät einfügen** im Projektnavigator, damit der Geräte-Auswahldialog geöffnet wird.
2. Wählen Sie unter **Gerät** SIMOTION D und klicken Sie auf die **Geräteausprägung** z. B. D435.

3. Wählen Sie die **SIMOTION Version** V4.2 und als **Optionsbaugruppe** CBE30.
   Die Optionsbaugruppe CBE30 können Sie in HW Konfig auch zu einem späteren Zeitpunkt noch einfügen (siehe Kapitel .

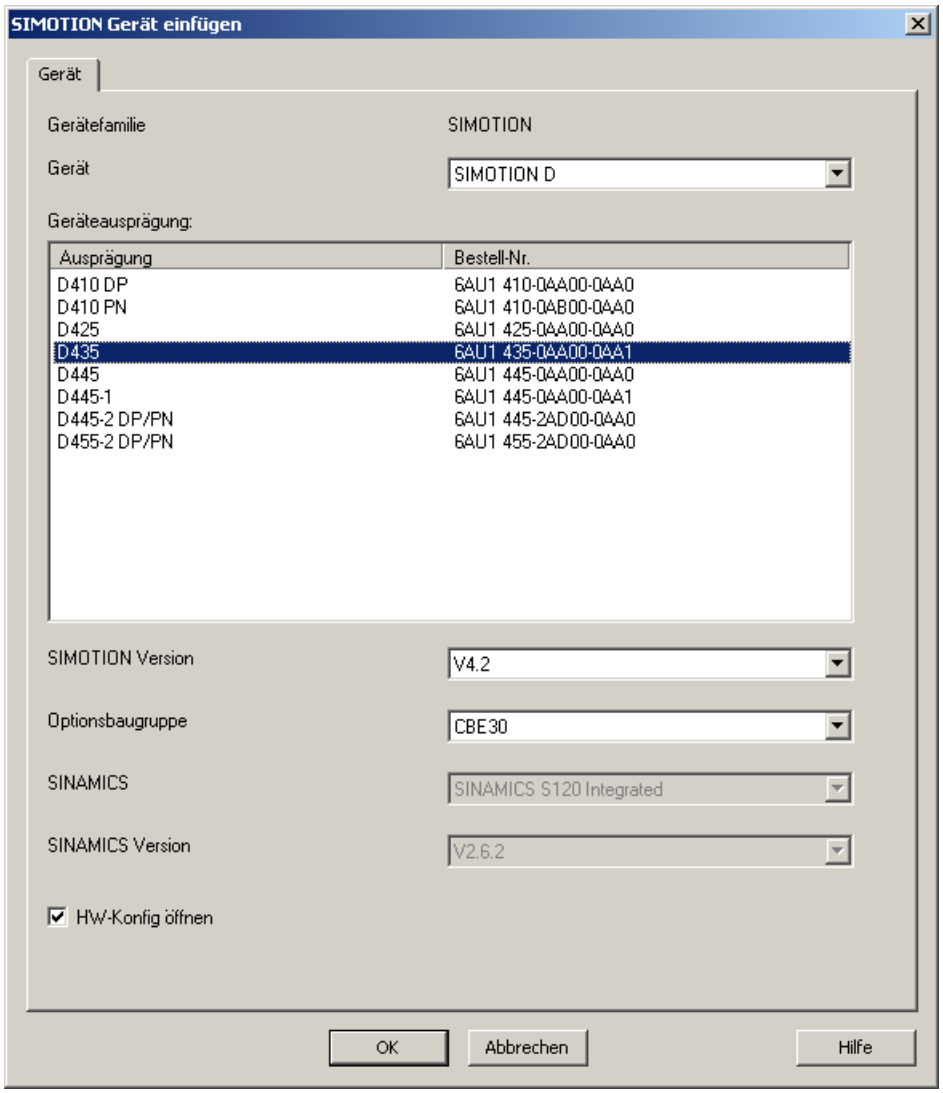


Bild 4-18 SIMOTION D435 mit CBE30 für PROFINET anlegen

4. Aktivieren Sie die Checkbox HW Konfig öffnen, um z. B. ein IO-System einzufügen.

5. Bestätigen Sie mit **OK**. Der Dialog zum Anlegen eines PROFINET-Subnetzes wird eingeblendet. Erstellen Sie mit Neu... ein neues Subnetz und tragen Sie die IP-Adresse und die Subnetzmaske. Bestätigen Sie mit OK.
6. Wählen Sie im folgenden Dialog die PG/PC-Schnittstelle aus und bestätigen Sie mit **OK**. Das Gerät wird eingefügt, HW Konfig wird geöffnet und es wird die Baugruppe mit dem projektierten PROFINET-Subnetz angezeigt.

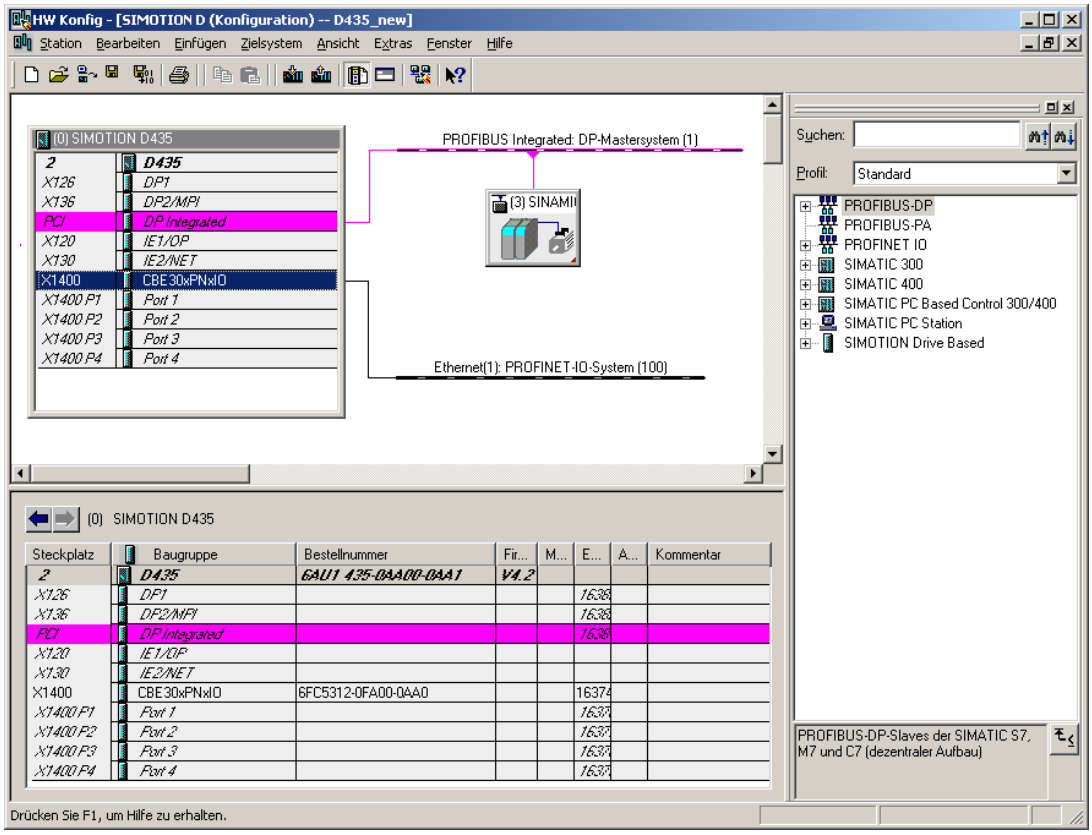


Bild 4-19 SIMOTION D435 in HW Konfig

Siehe auch

CBE30-PROFINET-Board einfügen und projektieren (Seite 82)

4.3.3.4 CBE30-PROFINET-Board einfügen und projektieren

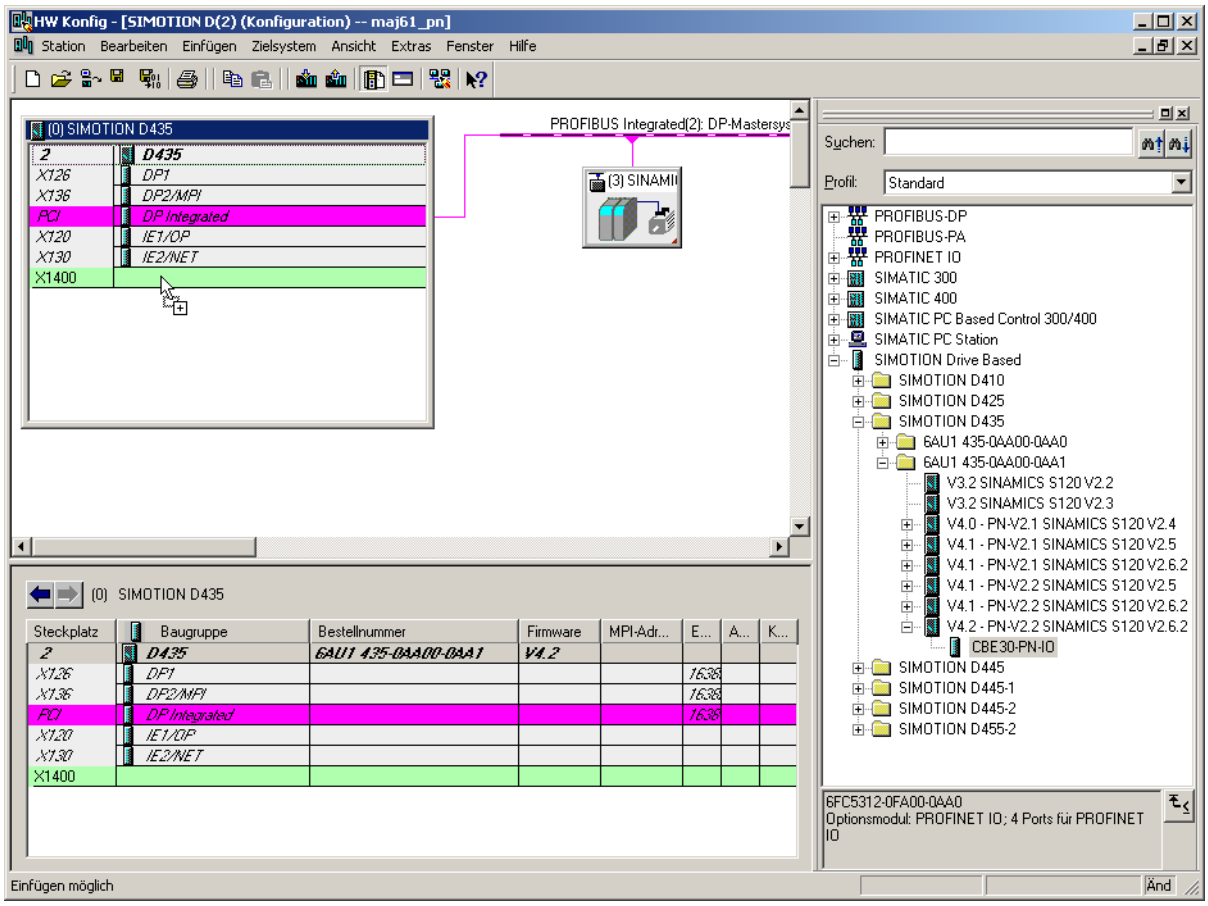
Voraussetzung

Sie haben bereits ein Projekt angelegt und ein SIMOTION D4x5 Gerät eingefügt.

Beim Gerät einfügen in SIMOTION SCOUT können Sie standardmäßig ein CBE30 einfügen. Die Optionsbaugruppe können Sie auch nachträglich in HW Konfig einfügen.

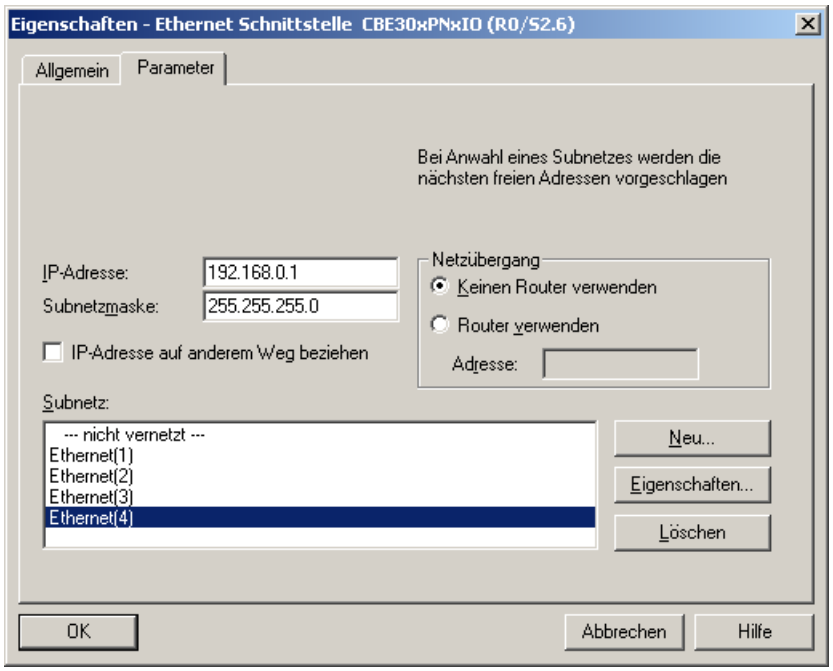
Vorgehensweise

1. Doppelklicken Sie im Projektnavigator auf die Baugruppe (hier eine D435). HW Konfig wird mit der entsprechenden Baugruppe eingeblendet.
2. Wählen Sie im Hardwarekatalog unter **SIMOTION Drive Based > SIMOTION D4x5 > 6AUxx > V4.2 - PN-V2.2xx** die geeignete Optionsbaugruppe. Beachten Sie den CPU-Typ und den Ausgabestand.
3. Klicken Sie auf die PROFINET-Baugruppe CBE30-PN. Sobald Sie die geeignete CBE30-PN auswählen, verfärbt sich X1400 im Baugruppenträger grün.



4. Ziehen Sie die CBE30-PN auf die entsprechende Schnittstelle der SIMOTION-Baugruppe (X1400). Es öffnet sich das Fenster **Eigenschaften - Ethernet Schnittstelle CBE30-PN (R0/S2.6).**
5. Klicken Sie auf **Neu** um ein neues Subnetz anzulegen. Der Dialog **Eigenschaften - Neues Subnetz Industrial Ethernet** wird eingeblendet.

6. Klicken Sie **OK**, um die Eingaben zu bestätigen. Es wird ein neues Ethernet Subnetz angelegt, z. B. Ethernet(4).



7. Wählen Sie das Subnetz aus.
8. Vergeben Sie die gewünschte IP-Adresse.
9. Übernehmen Sie die Einstellungen mit **OK**.

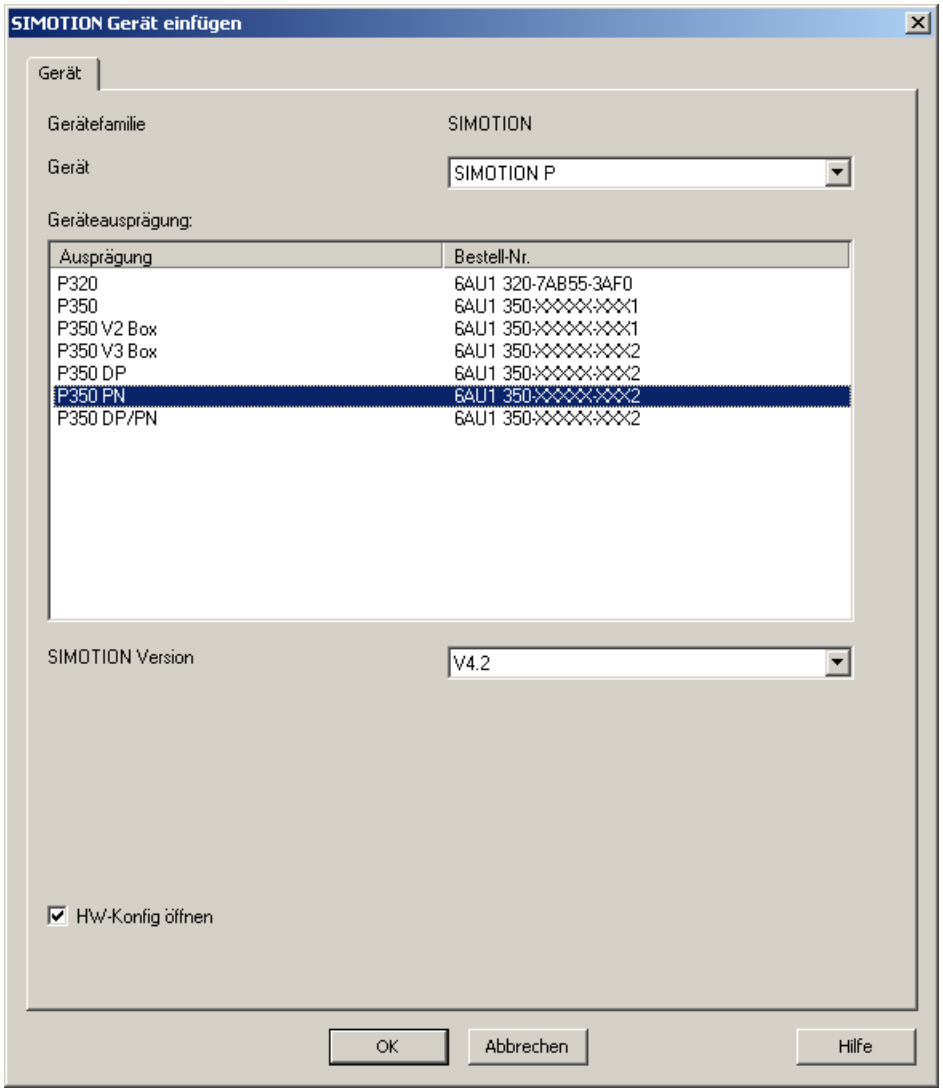
4.3.4 SIMOTION P einfügen und projektieren

Voraussetzung

Sie haben bereits ein Projekt angelegt und möchten eine SIMOTION P mit PROFINET einfügen. Am Beispiel einer SIMOTION P350 wird Ihnen die Vorgehensweise dargestellt.

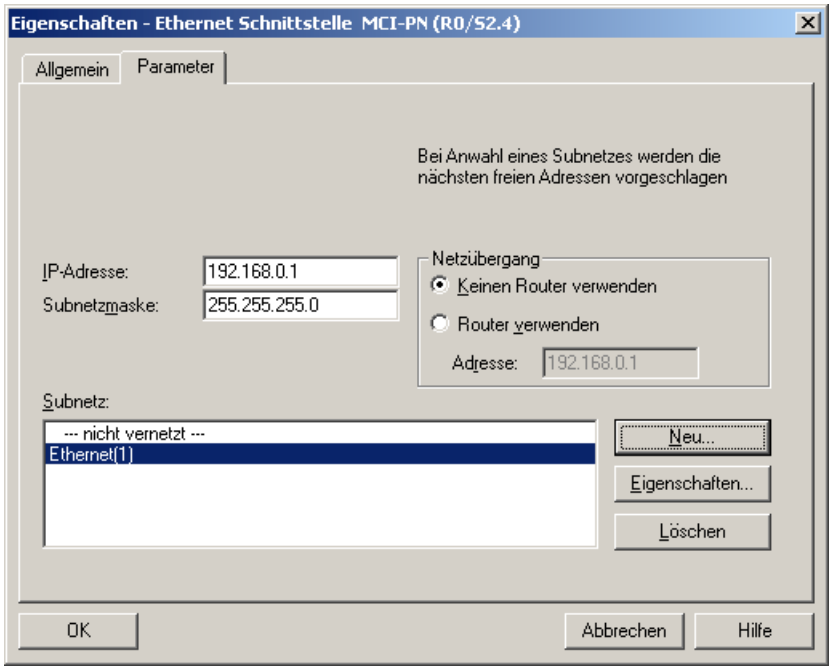
Vorgehensweise

1. Klicken Sie auf **SIMOTION Gerät einfügen**, um den Geräte-Auswahldialog zu öffnen.
2. Wählen Sie unter **Gerät** SIMOTION P, klicken Sie auf eine **Geräteausprägung** z. B. P350 PN und wählen Sie **SIMOTION Version** V4.2.



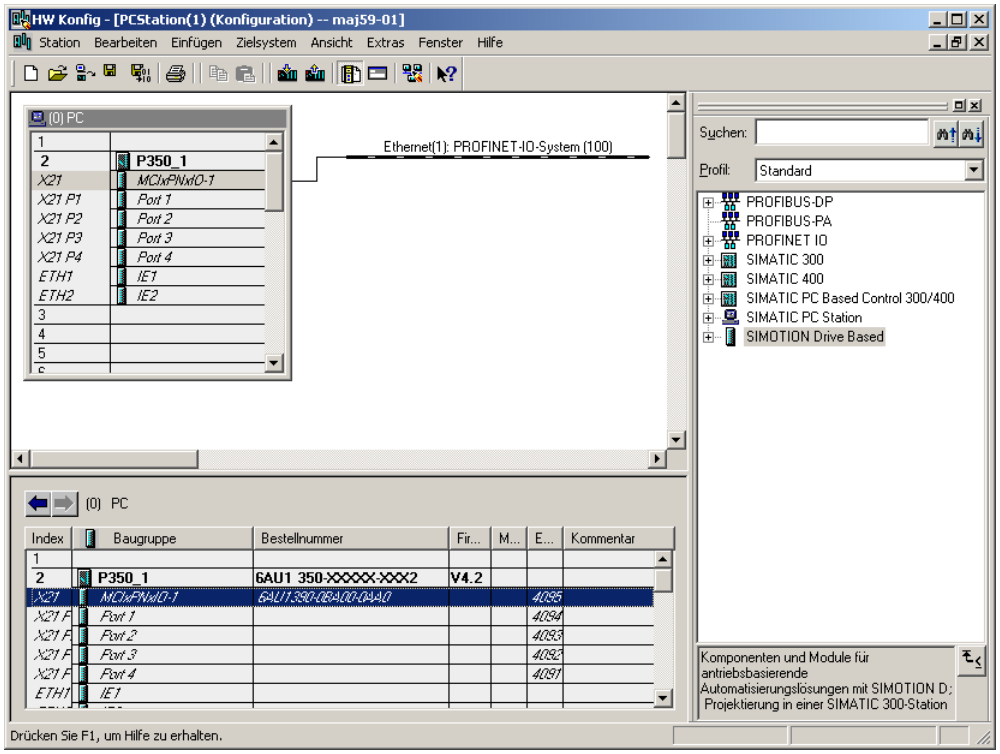
3. Aktivieren Sie die Checkbox **HW Konfig öffnen**, um z. B. ein IO-System einzufügen.
4. Bestätigen Sie mit **OK**. Der Dialog zum Anlegen eines PROFINET-Subnetzes wird eingeblendet.

5. Tragen Sie hier die IP-Adresse und die Subnetzmaske ein. Bestätigen Sie mit **OK**.



6. Wählen Sie im folgenden Dialog die PG/PC-Schnittstelle aus und bestätigen Sie mit **OK**.

   HW Konfig wird geöffnet und zeigt die Baugruppe mit dem projektierten PROFINET-Subnetz an.



4.3.5 SIMOTION C einfügen und projektieren

Voraussetzung

Sie haben bereits ein Projekt angelegt und möchten jetzt eine SIMOTION C mit PROFINET einfügen. Am Beispiel einer SIMOTION C240-PN wird Ihnen die Vorgehensweise dargestellt.

Vorgehensweise

1. Klicken Sie auf **SIMOTION Gerät einfügen**, um den Geräte-Auswahldialog zu öffnen.
2. Wählen Sie unter Gerät **SIMOTION C**, klicken Sie auf eine **Geräteausprägung** z. B. C240-PN und wählen Sie **SIMOTION Version** V4.2.

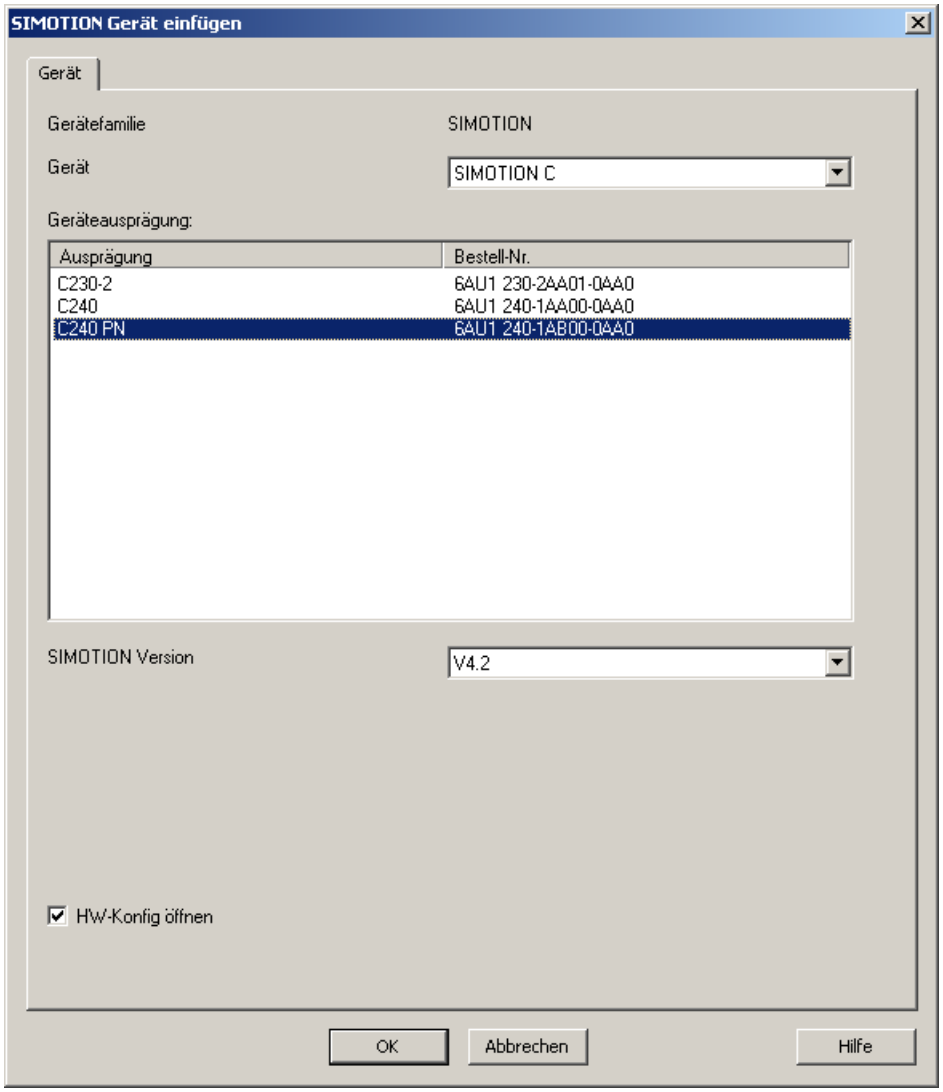


Bild 4-20 Neues Gerät C240 PN anlegen

3. Aktivieren Sie die Checkbox HW Konfig öffnen, um z. B. ein IO-System einzufügen

4. Der Dialog zum Anlegen eines PROFINET-Subnetzes wird eingeblendet. Geben Sie hier die IP-Adresse und die Subnetzmaske ein. Bestätigen Sie mit **OK**.

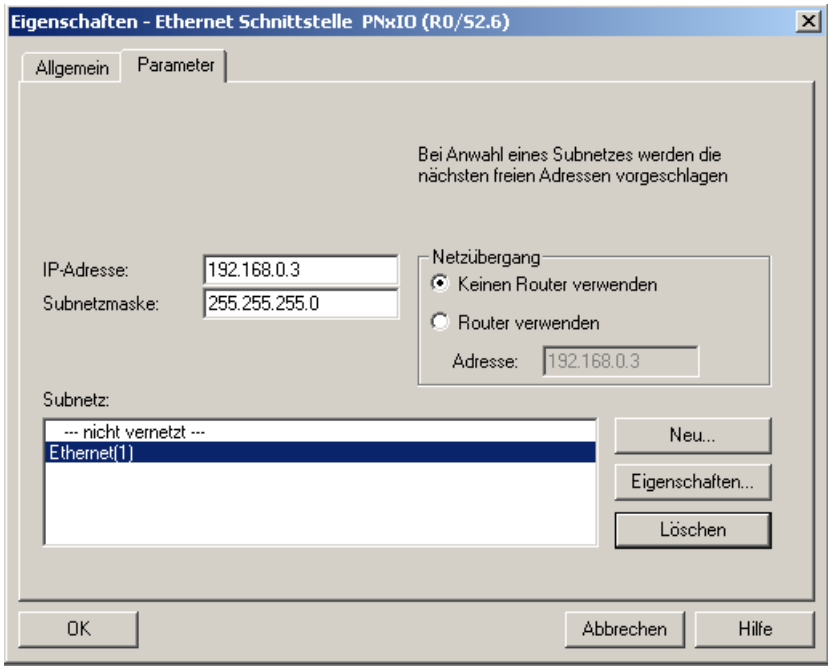


Bild 4-21 Neues Ethernet für C240 PN anlegen

5. Wählen Sie im folgenden Dialog die PG/PC-Schnittstelle aus und bestätigen Sie mit **OK**. HW-Konfig wird geöffnet und zeigt die Baugruppe mit dem projektierten PROFINET-Subnetz an.

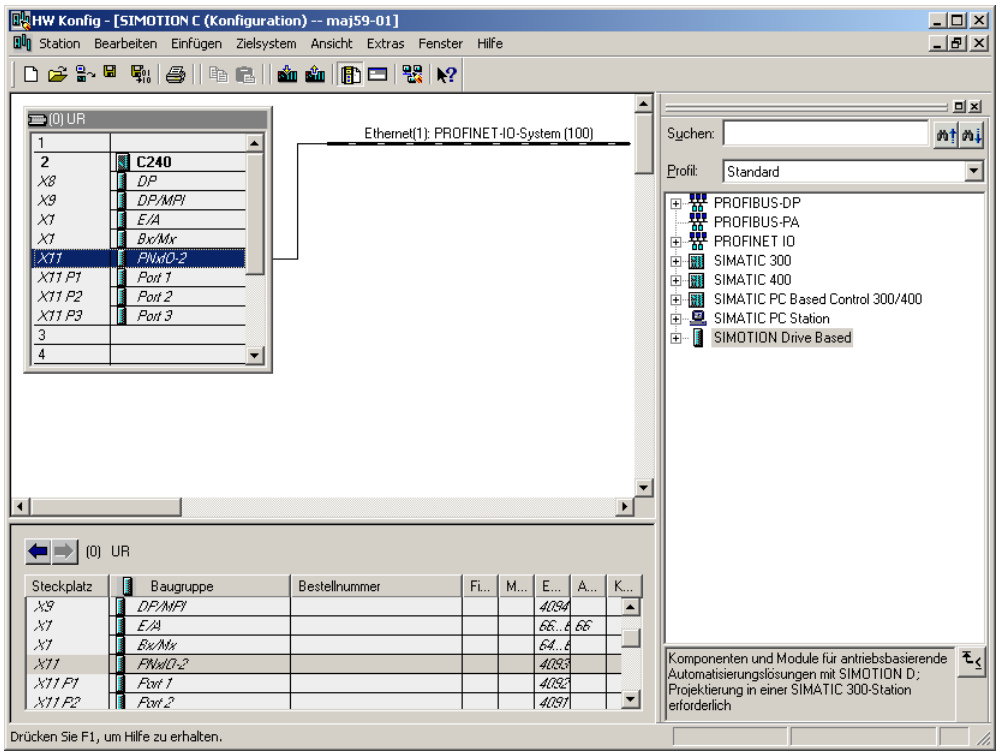


Bild 4-22 HW Konfig mit PROFINET für C240 PN

4.3.6 Sync-Domain anlegen

Eine Sync-Domain ist eine Gruppe von PROFINET-Geräten, die auf einen gemeinsamen Takt synchronisiert sind. Ein Gerät hat die Rolle des Sync-Masters (Taktgeber), alle anderen Geräte haben die Rolle eines Sync-Slaves.

Hinweis

Alle Geräte, die über IRT Daten austauschen, müssen einer Sync-Domain angehören und direkt miteinander verbunden sein, d. h., die Verbindung darf nicht durch nicht-IRT-fähige unterbrochen werden, da sonst die Synchronisationsinformationen nicht weitergeleitet werden.

Vorgehensweise

1. Öffnen Sie in HW Konfig die Station mit PROFINET-Geräten, die an der IRT-Kommunikation teilnehmen sollen und markieren Sie z. B. bei einer SIMOTION D455-2 DP/PN die PROFINET-Schnittstelle **PNxIO**.
2. Wählen Sie den Menübefehl **Bearbeiten > PROFINET IO > Domain Management**. Ein Dialogregister mit der Liste aller Geräte wird geöffnet. Eine Default Sync-Domain ist angelegt und die Geräte sind bereits zugeordnet.
3. Wählen Sie im oberen Feld die Station aus und doppelklicken Sie im unteren Feld auf das Gerät, das als Sync-Master projektiert werden soll, z. B. SIMOTION D. Der Eigenschaftsdialog des Gerätes wird geöffnet.

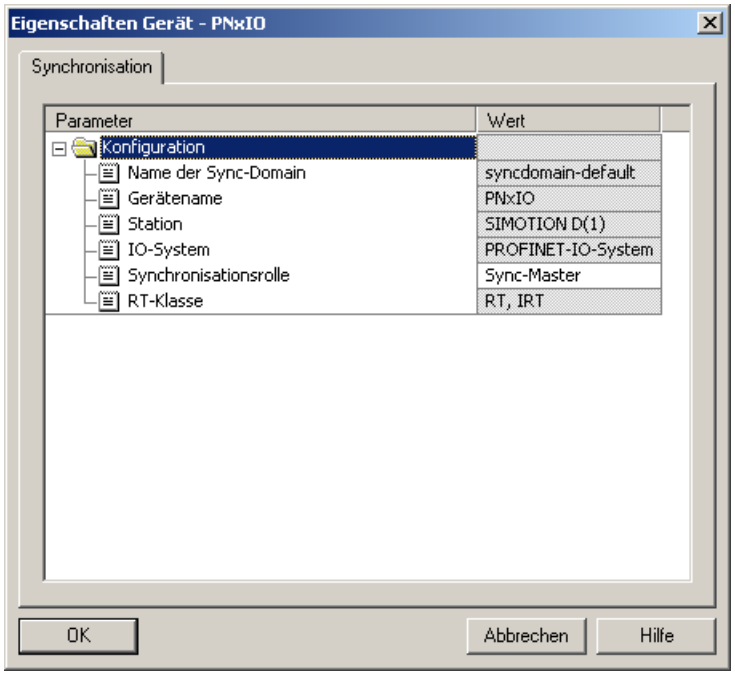


Bild 4-23 Synchronisierung auswählen

4. Stellen Sie die Synchronisationsart auf **Sync-Master** ein. IRT Hohe Performance ist die Standardeinstellung bei SIMOTION Controllern.

5. Quittieren Sie die Einstellungen mit **OK**.
6. Markieren Sie anschließend alle Geräte, die als Sync-Slaves projektiert werden sollen (Strg-Taste gedrückt halten und die Geräte nacheinander markieren).
7. Klicken Sie anschließend auf die Schaltfläche **Eigenschaften** Gerät.
8. Stellen Sie im Dialog die Synchronisationsart **Sync-Slave** ein.
9. Quittieren Sie die Einstellungen mit **OK**.

Hinweis

Alle Geräte, für die **nicht synchronisiert** ausgewählt ist, nehmen nicht an der IRT-Kommunikation, aber automatisch an der RT-Kommunikation teil.

4.3.7 Sendetakt und Aktualisierungszeiten festlegen

PROFINET RT und IRT ist eine zyklische Kommunikation, der Grundtakt ist der Sendetakt. Die Aktualisierungszeit ist ein Vielfaches (2^n) des Sendetakts und in diesem Takt werden die Geräte mit Daten versorgt. Die Aktualisierungszeit kann für jedes Gerät individuell eingestellt werden.

Hinweis

Bei IRT Hohe Performance werden die Geräte in jedem Sendetakt mit Daten versorgt (Sendetakt=Aktualisierungszeit). SIMOTION CPU unterstützen nur IRT Hohe Performance.

Für den SINAMICS kann jedoch über den Controller Applikationszyklus definiert werden, dass der SINAMICS nur alle n-Zyklen mit neuen Daten versorgt wird.

Vorgehensweise Sendetakt einstellen

1. Öffnen Sie in HW Konfig den Dialog **Domain Management**.

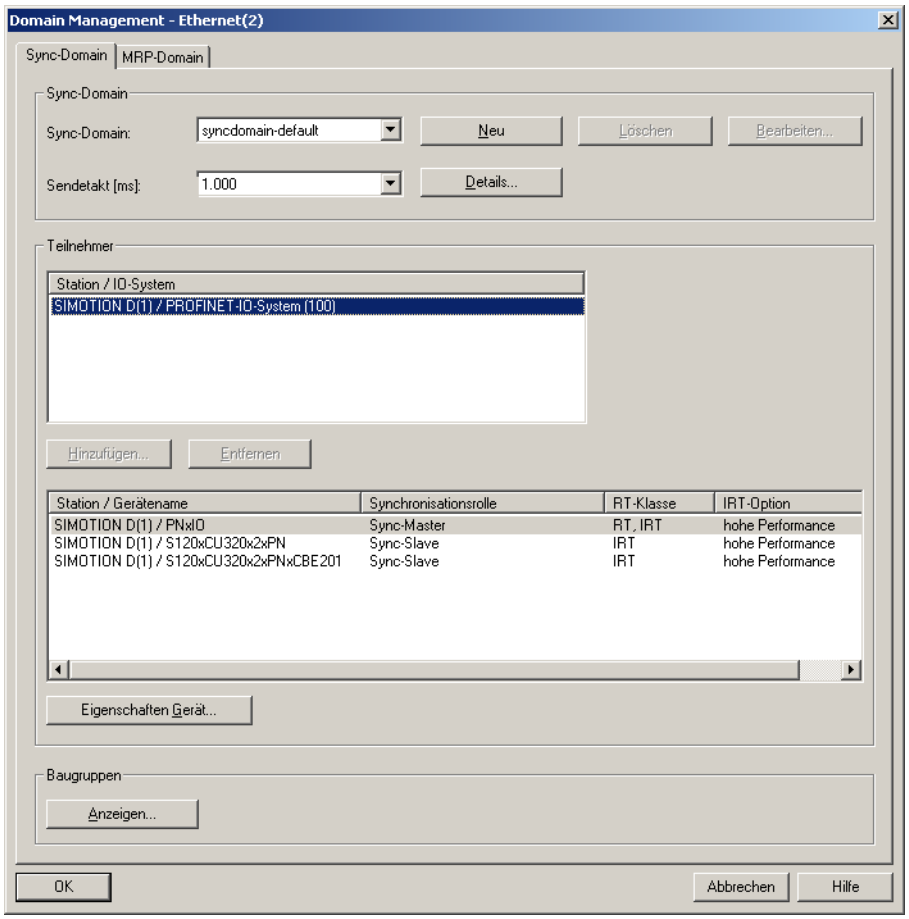


Bild 4-24 Domain Management

2. Wählen Sie einen an den Prozess angepassten Sendetakt. Der Sendetakt ist das kleinstmögliche Sendeintervall. Der Sendetakt ist auf 1 ms vor eingestellt.

Hinweis

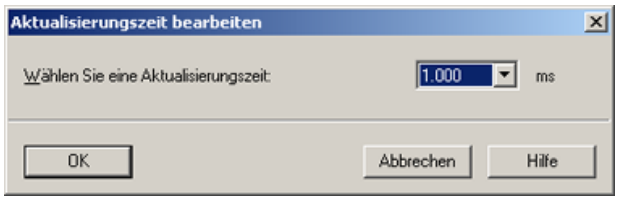
Es sollte nicht so schnell wie möglich, sondern nur so schnell wie nötig kommuniziert werden. Dies reduziert den Bandbreitenbedarf und entlastet die Geräte.

Aktualisierungszeiten für PROFINET IO mit PROFINET Geräten mit RT festlegen

Die Aktualisierungszeiten für den IO-Datenaustausch von PROFINET IO mit PROFINET Geräten mit RT wird im Dialog **Eigenschaften PROFINET IO-System** eingestellt.

1. Klicken Sie in HW Konfig auf den Strang des PROFINET IO-Systems und wählen Sie im Kontextmenü **Objekteigenschaften**. Der Dialog wird aufgeblendet.
2. Wechseln Sie in das Register **Aktualisierungszeit** und markieren Sie das Gerät in der Übersicht aller IO-Devices.

3. Klicken Sie auf **Bearbeiten**. Im Dialog **Aktualisierungszeit bearbeiten** können Sie die Aktualisierungszeit auswählen.



4. Bestätigen Sie mit **OK**.

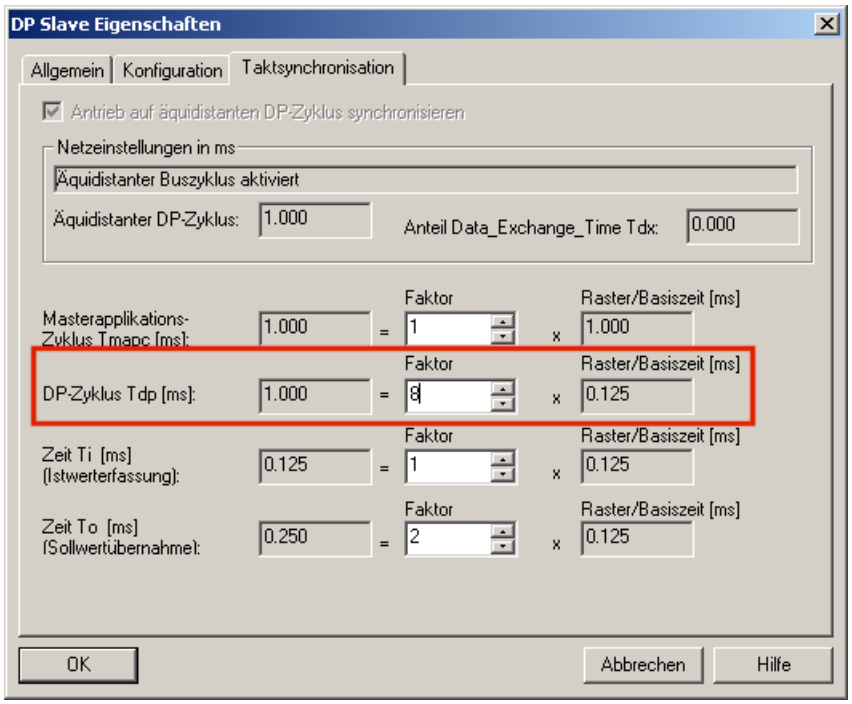
Verhältnis Sendetakt zum DP-Zyklus bei SINAMICS_Integrated (SIMOTION D) mit IRT Hohe Performance

Wird ein SINAMCIS Antrieb an einer SIMOTION D über PROFINET IO IRT Hohe Performance betrieben, muss der DP-Zyklus des integrierten PROFIBUS den Servo Zyklus entsprechen. Per Default ist der DP-Zyklus auf 3 ms eingestellt. Dies entspricht in der Regel nicht dem gewählten Servo-Takt bei PROFINET Applikationen. Entspricht der DP-Zyklus nicht dem Servo-Takt wird bei der Konsistenzprüfung im SCOUT eine Fehlermeldung generiert.

Den DP-Zyklus stellen Sie in den DP Slave Eigenschaften ein:

1. Doppelklicken Sie in HW Konfig auf den SINAMICS_Integrated. Das Fenster **DP Slave Eigenschaften** wird aufgeblendet.
2. Wechseln Sie in das Register **Taktsynchronisation** und aktivieren Sie die Checkbox Antrieb auf äquidistanten DP-Zyklus synchronisieren.

3. Stellen Sie den Faktor beim **DP-Zyklus Tdp** ein. Der DP-Zyklus muss gleich dem Servo Takt sein. Ist der Servo Takt z. B. 1 ms, müssen Sie als Faktor 8 eintragen (8 x [Basiszeit 0.125 ms] = 1 ms).



4. Bestätigen Sie mit **OK**.

Wenn PROFINET taktsynchron betrieben wird, muss der Servo-Takt immer dem PROFIBUS-Takt entsprechen. Servo-Takt und PROFIBUS-Takt können zum PROFINET-Takt untersetzt werden.

Beispiel:

PROFINET-Sendetakt = 0,5 ms

PROFIBUS-Takt = Servo-Takt = 1 ms

Der PROFIBUS-Takt kann zum PROFINET-Takt im Verhältnis 1:1 bis 1:16 betrieben werden.

4.3.8 Servo_fast, Taktuntersetzung zum Servo an der PROFINET-Schnittstelle

Taktuntersetzung zum Servo mit Servo_fast

Mit dem zweiten Servo-Takt können Sie zwei Bussysteme in unterschiedlichen Applikationstakten betreiben. Für beide Applikationstakte steht je ein zugeordneter Servo und IPO-Takt zur Verfügung. Damit können Sie Ihre Applikation in einen langsamen und einen schnellen Teil (Servo_fast und IPO_fast) aufteilen.

- Die Peripherie am schnellen Bussystem wird im schnellen Servo_fast/IPO_fast taktsynchron benutzt.
- Die Peripherie am langsamen Bussystem wird im langsamen Servo/IPO/IPO_2 taktsynchron benutzt.

Möglichkeiten der taktsynchronen Nutzung

Für die taktsynchrone Nutzung wird in HW Konfig an der I/O-Baugruppe der Applikationszyklus eingestellt:

- Bei DP-Slave wird der MACF (Master-Applikationszyklus) eingestellt. Wenn Sie zusätzlich PROFINET IO nutzen, wird nur MACF=1 unterstützt. Alle PROFIBUS Geräte laufen im langsamen Servo-Takt.
- Bei IO-Devices wird der CACF (Controller Application Cylce Factor) eingestellt. Ab V4.2 können Sie an IO-Devices zusätzlich den Servo-Takt auswählen (Serov_fast).

Folgende Einstellungen sind möglich:

Produktstufe	Eigenschaft	Applikationszyklus der Geräte am	
		PROFINET IO	PROFIBUS DP
Bis V4.2	1 Servo-Takt	Servo	Servo
V4.2	2 Servo-Takte	Servo_fast	Servo

Projektieren des Servo_fast am Beispiel einer D455-2 DP/PN

Als Vorrausetzung muss eine D455-2 DP/PN mit PROFINET-IO System und PROFIBUS DP projektiert sein.

1. Markieren Sie in HW Konfig die Baugruppe des SIMOTION Geräts z. B. D455 und wählen Sie im Menü **Bearbeiten > Objekteigenschaften**.
2. Aktivieren Sie im Register **Taktsynchrone Tasks** die Checkbox **Servo_fast/IPO_fast verwenden**.

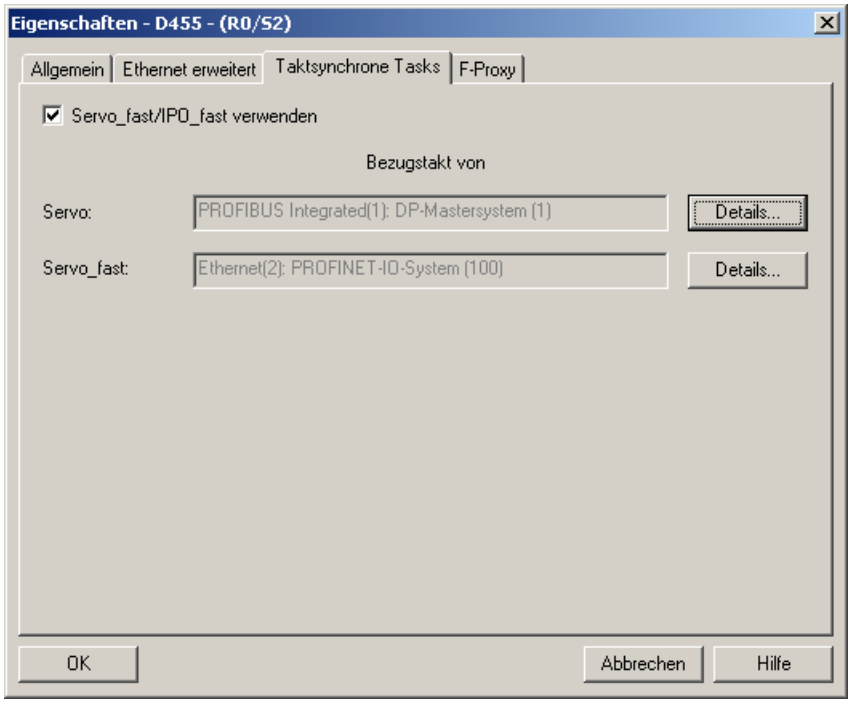


Bild 4-25 Servo_fast in HW Konfig projektieren

3. Klicken Sie neben Servo auf den Button **Details...** .
4. Tragen Sie im Register **Taktsynchronisation** des PROFIBUS DP einen Faktor für den DP-Zyklus Tdp ein z. B. Systemtakt PROFINET=1ms und PROFIBUS DP=4ms.

5. Bestätigen Sie mit **OK** und Speichern und übersetzen Sie das Projekt in HW Konfig.
6. Wechseln Sie in SIMOTION SCOUT und wählen Sie im Kontextmenü der SIMOTION CPU **Systemtakte einstellen**. Dort werden die Werte für den Servo und Servo_fast angezeigt, im Beispiel Taktverhältnis 1:4.

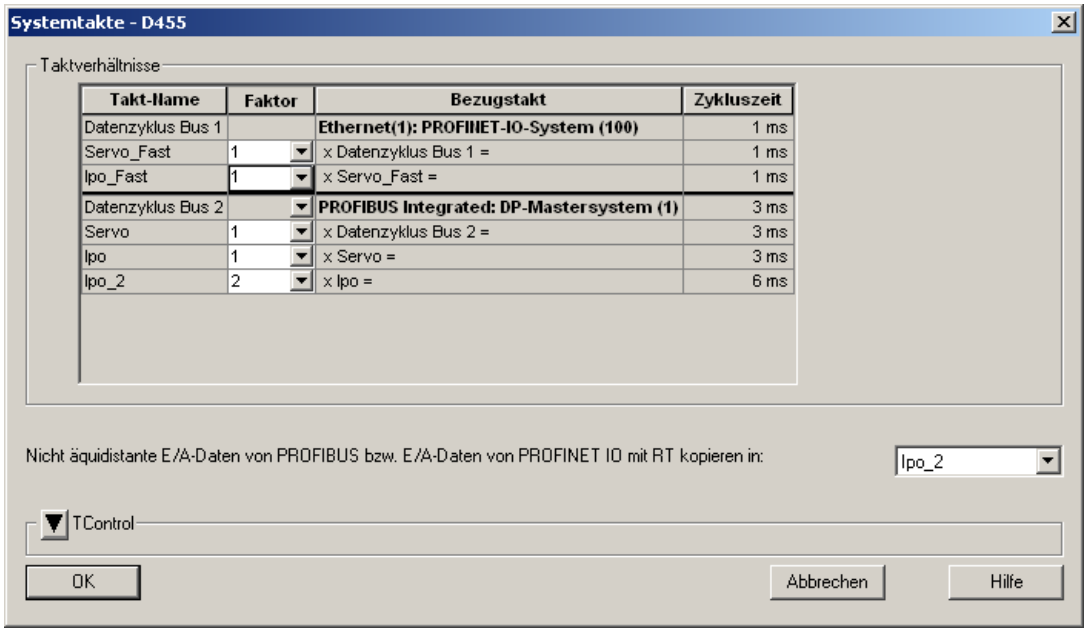


Bild 4-26 Systemtakte Servo und Servo_fast

Hinweis

Weitere Informationen finden Sie im Funktionshandbuch *SIMOTION SCOUT Basisfunktionen*.

4.3.9 Topologie projektieren

4.3.9.1 Topologie

Einleitung

Voraussetzung für IRT Hohe Performance ist die Topologieprojektierung und die Angabe, welches Gerät über welchen Port mit welchen anderen Geräten verbunden ist.

Hinweis

Bei IRT Hohe Flexibilität bzw. RT kann die Topologie optional projektiert werden, z. B. für die topologiebasierte Taufe oder für Diagnosezwecke.

Zur Festlegung der Eigenschaften von Leitungen zwischen den Ports der Switche haben Sie zwei Möglichkeiten:

Über den Topologie-Editor (Seite 99)

Über die Objekteigenschaften

4.3.9.2 Topologie-Editor (Grafische Ansicht)

Vorgehensweise

Mit dem Topologieeditor haben Sie eine Übersicht über sämtliche Ports im Projekt und können sie zentral verschalten.

Den Topologieeditor starten Sie mit dem Menübefehl **Bearbeiten > PROFINET IO > Topologie** in HW Konfig oder NetPro (PROFINET-Gerät muss ausgewählt sein).

Im Topologieeditor haben Sie zwei Möglichkeiten, die Topologie grafisch (ab STEP7 V5.4 SP2) oder tabellarisch anzuzeigen. Zum Verschalten ist die grafische Ansicht besser geeignet.

Beschreibung

Im Topologie Editor können Sie:

- Ports verschalten
- Eigenschaften der Verschaltung anpassen
- Passive Komponenten einfügen
- Im Online-Modus einen Vergleich Offline/Online anzeigen lassen

Vorgehensweise

1. Doppelklicken Sie im SCOUT auf die SIMOTION-Baugruppe um HW Konfig aufzurufen.
2. Wählen Sie die PROFINET-Baugruppe aus, z. B. PNxIO bei einer D445-2 DP/PN.
3. Führen Sie **Bearbeiten > PROFINET IO > Topologie** aus. Der Toplogie-Editor wird eingeblendet.

4. Klicken Sie auf **Grafische Ansicht**, um die Registerkarte in den Vordergrund zu schieben.

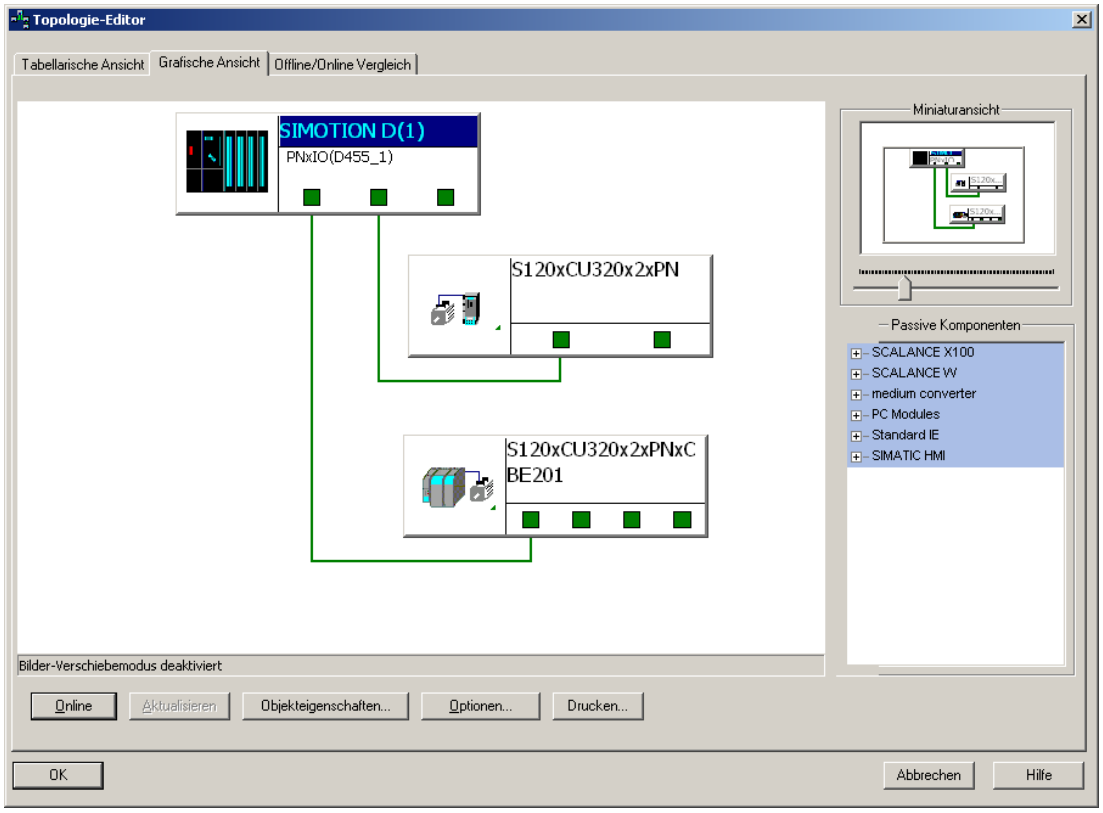
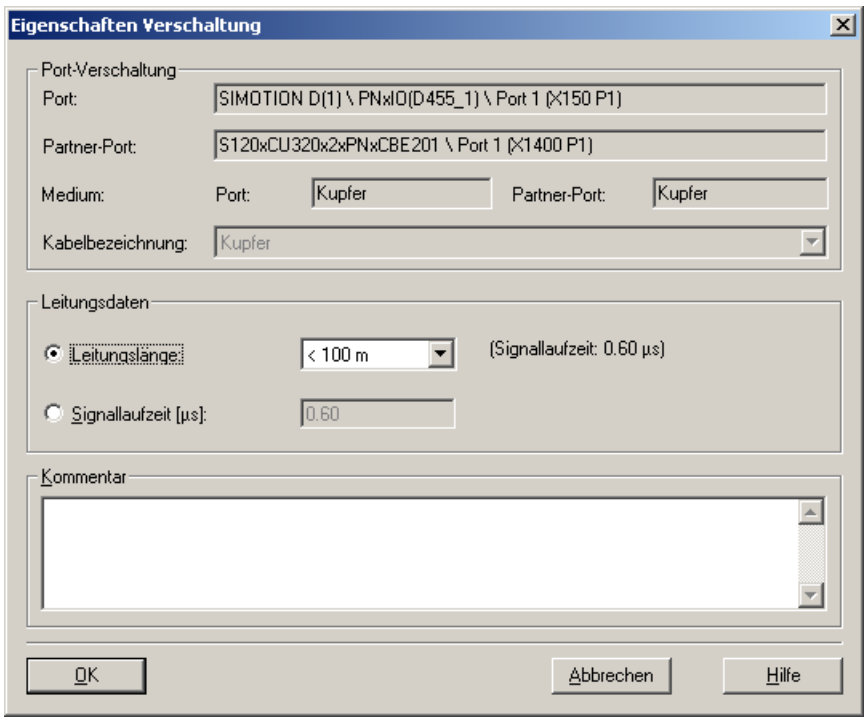


Bild 4-27 Topologie-Editor (grafische Ansicht)

5. Verbindungen zwischen Ports erstellen Sie, in dem Sie bei gedrückter linker Maustaste eine Verbindung zwischen den beiden Ports ziehen. Das Fenster **Eigenschaften Verschaltung** wird geöffnet.

6. Es wird Ihnen die Port-Verschaltung angezeigt. Sie haben die Möglichkeit die Leitungsdaten zu projektieren. Defaultmäßig wird eine Leitungslänge von <100m eingestellt, die empfohlen wird. Alternativ haben Sie die Möglichkeit die Signallaufzeit zu projektieren.



7. Bestätigen Sie mit **OK**.

Offline-/Online-Vergleich

Wenn Sie in den Online-Modus wechseln, wird die Topologie im Editor mit der realen Topologie verglichen. Nicht bekannte Komponenten werden mit Fragezeichen angezeigt, Verbindungen und Komponenten im RUN werden grün markiert.

4.3.9.3 Ports über den Topologie-Editor (tabellarische Ansicht) verschalten

Vorgehensweise

In der **Tabellarischen Ansicht** des Topologieeditors können Sie Ports verschalten.

Den Topologieeditor starten Sie mit dem Menübefehl **Bearbeiten > PROFINET IO > Topologie** in HW Konfig oder NetPro (PROFINET-Gerät muss ausgewählt sein).

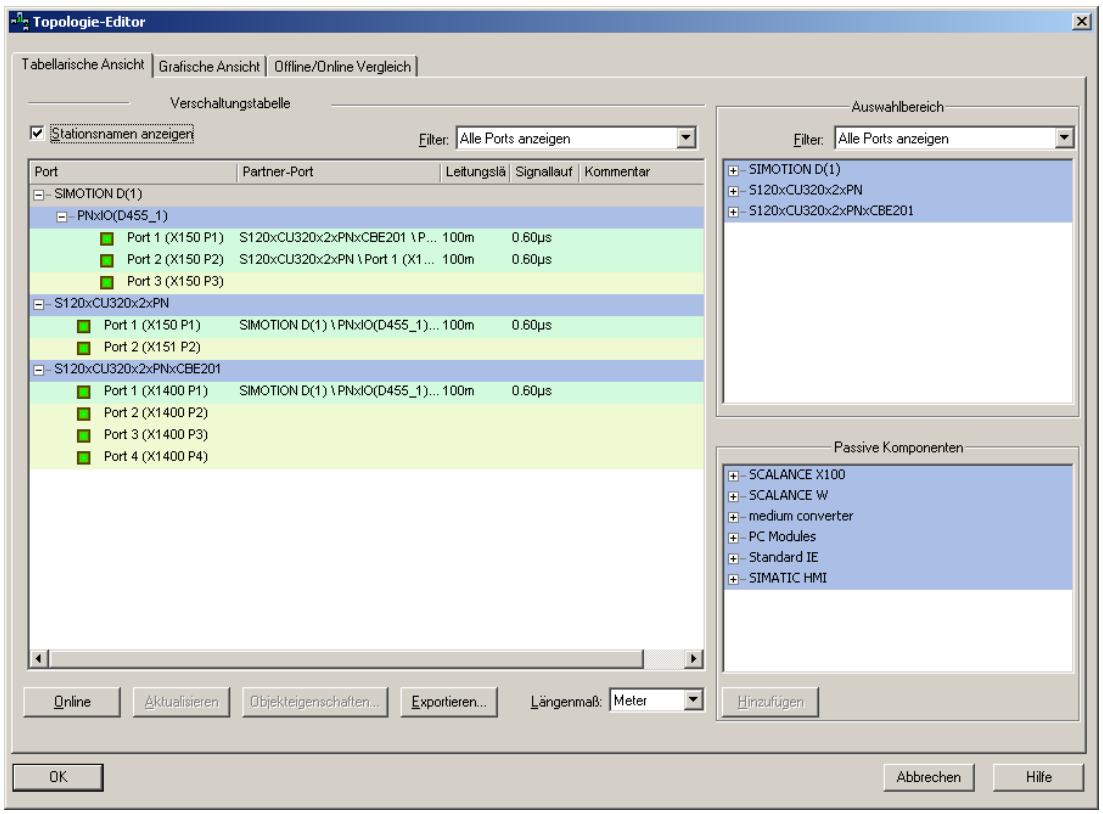


Bild 4-28 Topologie Editor

In der Verschaltungstabelle auf der linken Seite werden alle projektierten PROFINET IO Geräte mit ihren Ports aufgelistet. Über den Filter-Dialog können Sie wählen, ob alle Ports, nur die noch nicht verschalteten Ports oder nur die bereits verschalteten Ports angezeigt werden sollen.

Verschaltung von Ports in der tabellarischen Übersicht

1. Zur Verschaltung von Ports unterschiedlicher Geräte wählen Sie im rechten Auswahlbereich den Port eines Gerätes aus, den Sie verschalten möchten.
2. Ziehen Sie diesen Port auf den gewünschten Port eines Gerätes in der Verschaltungstabelle. Darauf öffnet sich der Dialog Eigenschaften Verschaltungen.
3. Projektieren Sie die Leitungsdaten. Defaultmäßig sollte eine Leitungslänge von <100m eingestellt werden.
4. Bestätigen Sie Ihre Eingabe mit **OK**.

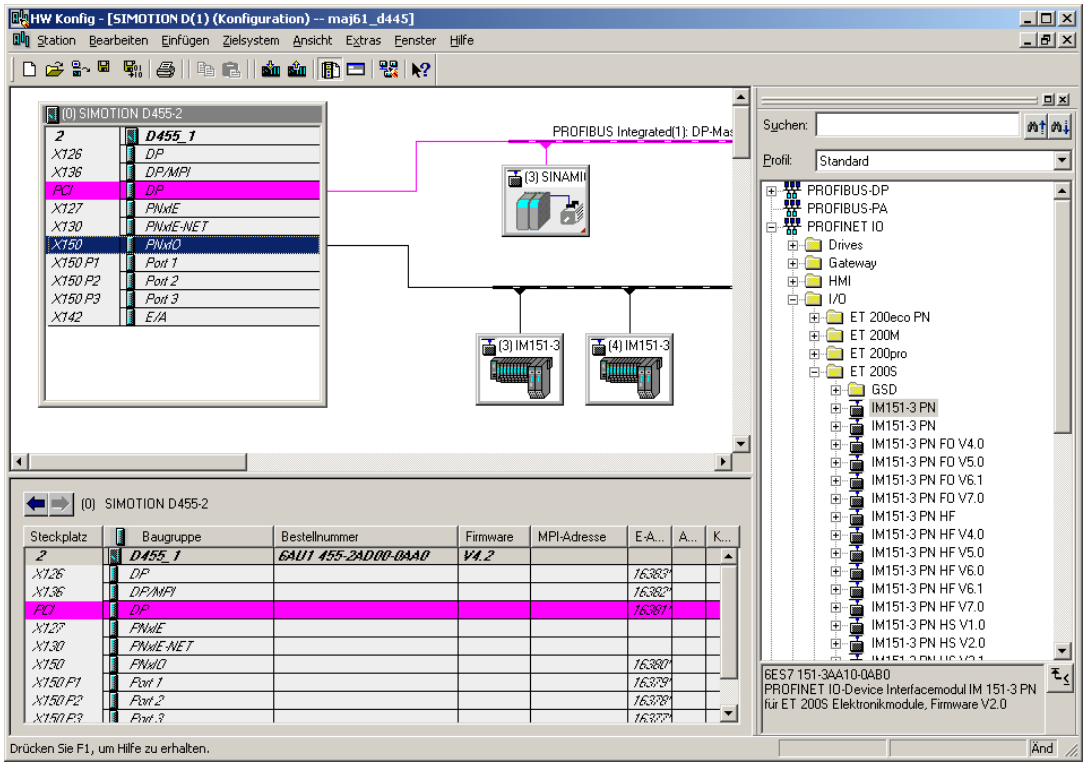
4.3.10 IO-Device anlegen

Voraussetzung

Sie haben bereits ein PROFINET IO-System angelegt und eine PROFINET IO-Baugruppe, z. B. SIMOTION D455-2 PN, projektiert (siehe SIMOTION D4x5-2/D410 PN einfügen und projektieren (Seite 77)).

Vorgehensweise bei PROFINET IO-Devices über den Hardware Katalog

1. Doppelklicken Sie in SIMOTION SCOUT auf die entsprechende Baugruppe um HW Konfig zu öffnen.
2. Wählen Sie im Baugruppenkatalog unter PROFINET IO die Baugruppe aus, die Sie an das PROFINET IO-System anschließen möchten.
3. Ziehen Sie die Baugruppe auf den Strang des PROFINET IO Systems. Das IO-Device wird eingefügt.



4. Speichern und übersetzen Sie die Einstellungen in HW Konfig.

Vorgehensweise bei PROFINET IO-Devices von Fremdherstellern

1. Doppelklicken Sie auf die entsprechende Baugruppe um HW Konfig zu öffnen.
2. Wählen Sie den Menübefehl **Extras > GSD-Dateien** installieren.
3. Wählen Sie im Dialog **GSD-Dateien installieren** die zu installierende GSD-Datei aus.

4. Klicken Sie auf den Button **Installieren**.
5. Schließen Sie den Dialog durch Klicken des Buttons **Schließen**.
6. Wählen Sie im Baugruppenkatalog unter PROFINET IO die Baugruppe aus, die Sie an das PROFINET IO-System anschließen möchten.
7. Ziehen Sie die Baugruppe auf den Strang des PROFINET IO System. Das IO-Device wird eingefügt.
8. Speichern und übersetzen Sie die Einstellungen in HW Konfig.

4.3.11 SINAMICS S120 einfügen und projektieren

Voraussetzung

Sie haben in Ihrem Projekt eine SIMOTION D4x5-2 mit integrierter PROFINET-Schnittstelle eingefügt, es ist bereits ein PROFINET IO Subnetz angelegt und der Sync-Master ist projektiert.

Hinweis

Die Projektierung ist nahezu identisch für alle SINAMICS S120 und SIMOTION D Geräte.

Vorgehensweise

1. Wählen Sie im Hardware-Katalog von HW Konfig den Eintrag **PROFINET IO > Drives > SINAMICS** und dort die Baugruppe, z. B. **SINAMICS S120 CU320-2 PN**.
2. Klicken Sie auf den Eintrag und ziehen den Antrieb z. B. **V4.4** auf das PROFINET IO Subnetz. Es öffnet sich das Fenster **Eigenschaften - Ethernet Schnittstelle SINAMICS-S120xCU320x2xPN**.
   Dort ist bereits eine IP-Adresse vorgeschlagen und das Subnetz ausgewählt.
3. Klicken Sie auf **OK**, um die Einstellung zu übernehmen.
4. Wählen Sie in HW Konfig **Station > Speichern und Änderungen übersetzen**.

Telegramm einstellen in SIMOTION SCOUT ab V4.2

Für die taktsynchrone Kommunikation muss ein Telegramm projektieren werden (z. B. Telegramm 105), das eine Aufsynchronisierung des Antriebs auf PROFINET ermöglicht. Das Telegramm wird automatisch im SCOUT bei der Konfiguration des Antriebsgerätes und nach der Zuweisung einer Achse eingestellt.

1. Klicken Sie im SCOUT im Projektnavigator und dem Antriebsgerät auf **Antriebsgerät konfigurieren**, falls Sie noch keine Einspeisung und Antrieb projektiert haben.
2. Durchlaufen Sie den Assistenten zum Konfigurieren des Antriebsgerätes.

3. Nachdem Sie den Assistenten abgeschlossen haben, klicken Sie im Projektnavigator unterhalb des Antriebsgeräts auf **Kommunikation > Telegrammkonfiguration**. Im Register **IF1: PROFIdrive PZD-Telegramme** sind die Telegramme aufgelistet. Ein Telegramm ist nach der Antriebsinbetriebnahme noch nicht projektiert.
4. Fügen Sie daher ein neues TO Achse ein und durchlaufen Sie den Achsassistenten. Im Assistent verschalten Sie die Achse auf das entsprechende Antriebsobjekt des S120 und automatisch wird ein entsprechendes Telegramm angelegt (Symbolische Zuordnung).
5. Wählen Sie im Menü **Projekt > Speichern und alles übersetzen**. Damit werden bei der symbolischen Zuordnung die Adressen automatisch eingerichtet. Über den Menüeintrag **Projekt > Adressen einrichten** können Sie ebenso die symbolische Zuordnung der Adressen anstoßen.
6. Nachdem die Achskonfiguration abgeschlossen ist, klicken Sie im Projektnavigator unterhalb des Antriebsgeräts auf **Kommunikation > Telegrammkonfiguration**. Im Register **IF1: PROFIdrive PZD-Telegramme** ist nun beim Antrieb das Telegramm aufgeführt (z. B. SIEMENS Telegramm 105).

Telegramm in HW Konfig

Alternativ können Sie das Telegramm auch in HW Konfig zuweisen. Dazu müssen das Antriebsgerät und der Antrieb im SCOUT schon projektiert sein. Dann können Sie in HW Konfig wie folgt vorgehen.

Hinweis

Für die Telegrammverschaltung wird die Symbolische Zuorndung in SIMOTION SCOUT empfohlen. Beachten Sie dazu, dass im Menü bei **Projekt > Symbolische Zuordnung verwenden** ein Häkchen gesetzt ist.

Wenn Sie symbolische Zuordnung verwenden, findet die Verwaltung des Telegramms zwischen SIMOTION und den Antriebs-DOs vollständig unter Kontrolle des SCOUT statt. D.h. SCOUT erzeugt eigenständig Telegramme, die alle gemäß der technologischen Projektierung notwendigen Signale enthalten. Die Platzierung der Signale im Telegramm und die Telegrammgröße werden hierbei automatisch von SCOUT verwaltet und können nicht mehr vom Anwender beeinflusst bzw. geändert werden.

1. Wählen Sie den eingefügten SINAMICS-Antrieb aus und doppelklicken Sie in der unteren Tabelle beim Antrieb auf den Eintrag **SIEMENS / Standard Telegramm xx**.
   Der Dialog **Eigenschaften SIEMENS / Standard Telegramm xx** wird aufgerufen.
2. Wählen Sie das entsprechende Telegramm aus. Nach dem Speichern kann das Telegramm dann auch im SCOUT im Projektnavigator unter **<"Antriebsgerät_xx"> - Kommunikation > Telegrammkonfiguration** ausgewählt werden. Ein Abgleich mit HW Konfig ist möglich.

Einstellungen für Taktsynchronität (ab V4.2)

Für Taktsynchronität muss sich das Antriebsgerät auf den Takt des PROFINET aufsynchronisieren. Diese Einstellungen nehmen Sie an der PN-Schnittstelle des Antriebsgeräts und am SIMOTION Gerät (Sync-Master) vor.

Einstellungen am SIMOTION Gerät (Sync-Master)

1. Markieren Sie in HW Konfig die Baugruppe des SIMOTION Geräts z. B. D455 und wählen Sie im Menü **Bearbeiten > Objekteigenschaften**.
2. Klicken Sie im Register Taktsynchrone Tasks auf den Button **Details...** .
3. Wählen Sie im Dialog **Details für Servo** unter **Ti/To-Modus** die Einstellung **automatisch**. Die Berechnung der Takte und Zeitkonstanten für alle IO-Devices (Sync-Slaves) am PROFINET werden dann automatisch vom System durchgeführt.

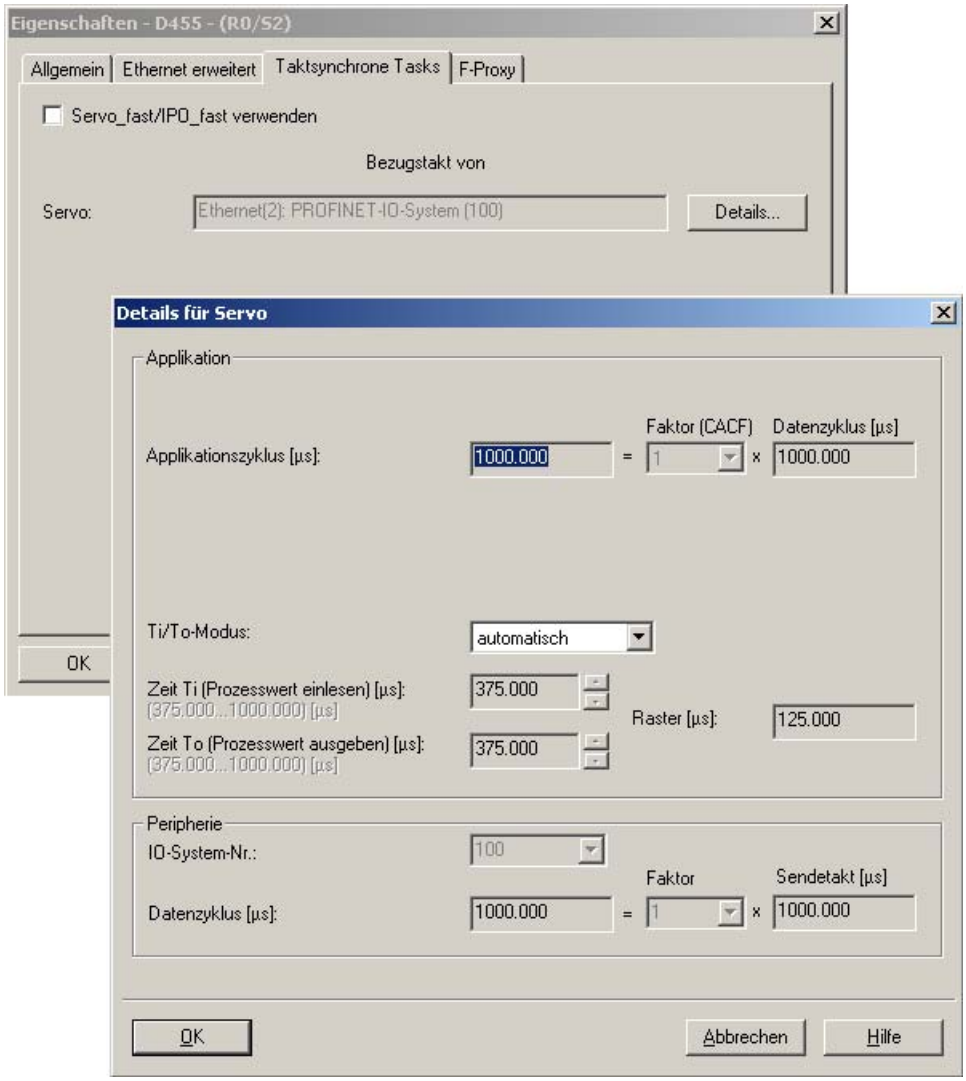


Bild 4-29 Taktsynchrone Task am SIMOTION Gerät projektieren.

4. Schließen Sie die beiden Dialoge mit **OK**.

Einstellungen am SINAMICS Antrieb (Sync-Slave)

1. Wählen Sie den SINAMICS Antrieb am PROFINET IO System aus und doppelklicken Sie in der unteren Tabelle auf den Eintrag des PROFINET Interfaces, z. B. PN-IO.
   Der Dialog **Eigenschaften PN--IO** wird eingeblendet.
2. Wählen Sie im Register **IO-Zyklus** unter **IO-Device taktsynchron zuordnen** den Eintrag **Servo**. Der Antrieb wird taktsynchron betrieben. Die Zeitkonstanten werden automatisch berechnet.

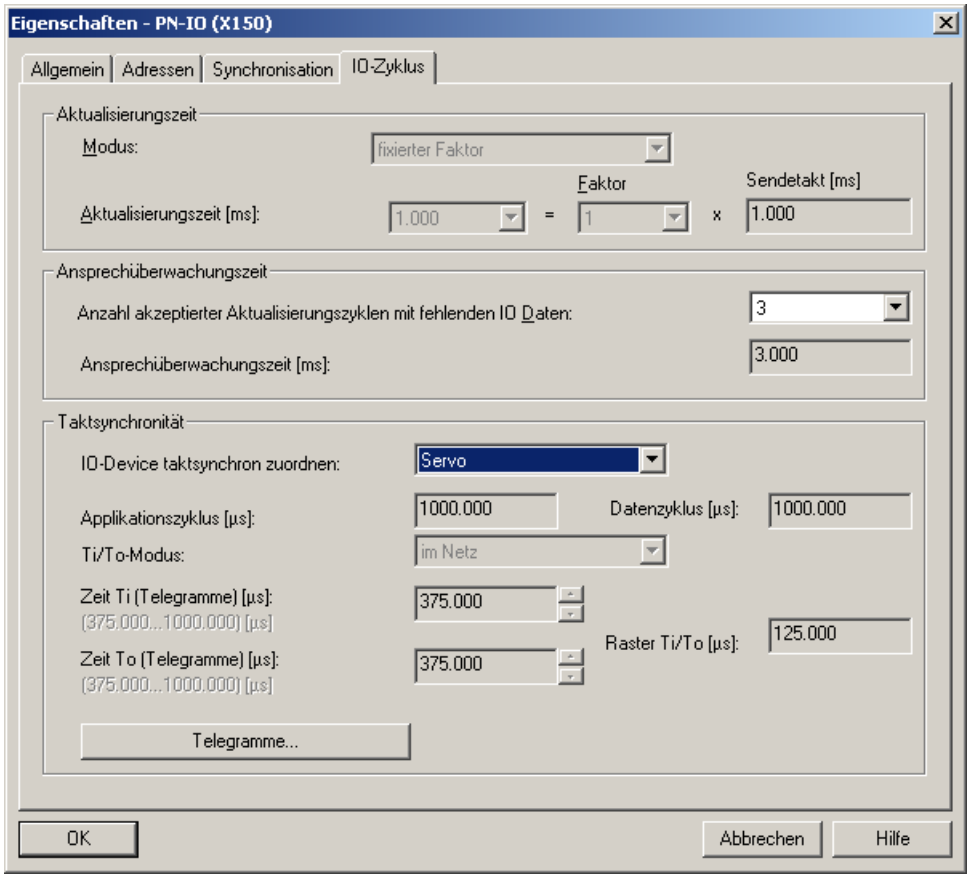


Bild 4-30 Taktsynchrone Task beim PROFINET IO-Device projektieren

3. Unter **Applikationszyklus** wird Ihnen der aktuelle Takt angezeigt. Um eine Taktuntersetzung zu projektieren, müssen Sie im **Ablaufsystem** unter **Systemtakte einstellen** die Untersetzung projektieren. Dies ist z. B. notwendig, wenn der Servo langsamer als der IRT-Takt ist. Der Servo muss jedoch im IRT-Takt fertig berechnet sein.

4. Wechseln Sie gegebenenfalls in das Register **Synchronisation**, um die **Synchronisationsart**, hier **Sync-Slave** auszuwählen. Das kann auch unter **Domain Management** eingestellt werden.
5. Bestätigen Sie die Eingaben mit **OK**.

Hinweis

Als weitere Arbeitsschritte müssen Sie den Antrieb in die Sync-Domain einbinden (siehe Sync-Domain anlegen), die IP-Adresse in den Antrieb laden, siehe Gerätenamen und IP-Adressen für IO-Devices vergeben (Seite 108)und die Ports verschalten (siehe Ports über den Topologie-Editor (tabellarische Ansicht) verschalten (Seite 99)).

Siehe auch

Sync-Domain anlegen (Seite 89)

4.3.12 IP-Adresse und Kommunikationsname

Einleitung

Ein Gerät bzw. Controller besitzt eine feste MAC-, eine projektierbare IP-Adresse und einen Kommunikationsnamen. Die IP-Adresse, Subnetzmaske und den Kommunikationsnamen legen Sie im Engineering unter den Eigenschaften der Ethernet Schnittstelle fest. Damit besitzen die Geräte im Projekt eine eindeutige Zuordnung.

Den Kommunikationsnamen (Gerätenamen) finden Sie Dialog Eigenschaften - Schnittstelle, wenn Sie auf das Gerät bzw. den Controller doppelklicken. Klicken Sie im Dialog auf **Eigenschaften**, um die IP-Adresse festzulegen. Siehe auch dazu CBE30-PROFINET-Board einfügen und projektieren (Seite 82) .

Richtlinien für Kommunikationsnamen

Beim Device ist der Kommunikationsname gleich dem Gerätenamen, z. B. *Drive1* bei einem Antrieb. Bei Controllern ist der Kommunikationsname der Name des PROFINET-Interfaces, z. B. *CBE30xPNxIO* bei einer CBE30 an SIMOTION D.

Kommunikationsnamen unterliegen bestimmten Syntaxregeln, d. h., sie müssen grundsätzlich DHCP-konform sein und für SIMOTION und SIMATIC existieren noch weitere Randbedingungen.

- Buchstaben und Ziffer sind zulässig a-z und 0-9
- keine Sonderzeichen ! " § $ % & / ( ) = ? * ' _ : ; > < , # + | ~ \ }] [{

- Der Name kann sich aus mehreren Teilen zusammensetzen
  - Label.Label.Label.Label
  - Der Punkt dient als Trennzeichen
  - Das Label muss mit einem Buchstaben beginnen bzw. enden
  - Die maximale Länge des Labels sind 63 Zeichen
  - Label darf nicht mit "xn-" beginnen
- Die maximale Gesamtlänge des Namens beträgt 240 Zeichen
- Reservierte Namen "port-xyz" oder "port-xyz-abcde"
- Es wird nicht zwischen Groß- und Kleinschreibung unterschieden. Da vom Engineering-System alle Namen mit Kleinbuchstaben angezeigt werden, sollten Sie auch nur diese verwenden.
- Im Zusammenhang mit SCOUT darf das Minus "-" nicht verwendet werden
- Nicht zulässige Zeichen werden vom Engineering durch ein x ersetzt.

Taufe von Controller und Devices im Online-Modus

Der Controller und die Devices besitzen im Auslieferungszustand noch keinen Kommunikationsnamen und keine IP-Adresse, diese müssen erst zugewiesen werden. Bei der Adressvergabe, d. h. IP-Adresse und Kommunikationsname zuweisen, auch Taufe genannt, wird zwischen Controllern und Devices unterschieden. Devices und Controller können Sie auf unterschiedliche Arten taufen.

Controller Taufe

- Download der Applikation
- Engineering Software
  - HW Konfig, NetPro, SCOUT
  - Primary Setup Tool (PST)
- Durch die Applikation (Systemfunktion _setNameofStation für SIMOTION)

Hinweis

Bei der Taufe mit der Engineering Software sollten Sie sich gerade bei größeren Anlagen direkt mit dem Gerät verbinden, da dadurch das zu taufende Gerät eindeutig identifiziert wird. Alternativ steht das Feature Blinken zur Verfügung, in dem das Gerät über eine blinkende LED identifiziert werden kann.

Device Taufe

- Engineering Software
  - HW Konfig, NetPro
  - SCOUT, Starter
  - Primary Setup Tool (PST)
- Vorher die MMC oder CF-Karte beschreiben und dann stecken

Topologiebasierte Taufe für Device

Devices können auch ohne MMC- bzw. CF-Karte getauft werden. Dies wird als topologiebasierte Taufe bezeichnet. Dies wird nur ab bestimmten Softwareständen unterstützt.

- SIMATIC S7-300 FW ≥ V 2.7
- SIMATIC S7-400 FW ≥ V 5.2
- SIMOTION FW ≥ V4.1.2 mit PN V2.2
- SINAMICS FW ≥ V2.5.1.10 mit PN V2.2
- ET200S FW ≥ 6.0
- ET200S HS FW ≥ 2.0

4.3.13 Gerätenamen und IP-Adressen für IO-Devices vergeben

Einleitung

Um auf ein Gerät (Controller oder Device) mit dem Engineering online gehen zu können, muss dem Gerät im Engineering eine IP-Adresse zugewiesen werden. Weiterhin muss den Geräten ein im Netzwerk eindeutiger Kommunikationsname zugewiesen werden. Dieser wird benötigt, damit der Controller seine ihm zugeordneten Devices identifizieren kann.

Die IP-Adresse können Sie im Dialog **Eigenschaften - Ethernet Schnittstelle …** festlegen (kann mit Doppelklick auf das Gerät geöffnet werden). Außerdem wird standardmäßig ein Gerätename eingetragen, den Sie ändern können. Als Standardeinstellung ist die Einstellung **IP-Adresse durch Controller zuweisen** aktiv. D. h., im Hochlauf identifiziert der Controller die ihm zugeordneten Devices über den Kommunikationsnamen und weist ihnen dann die im Engineering festgelegte IP-Adresse zu. Es wird empfohlen, diese Funktion nicht zu deaktivieren.

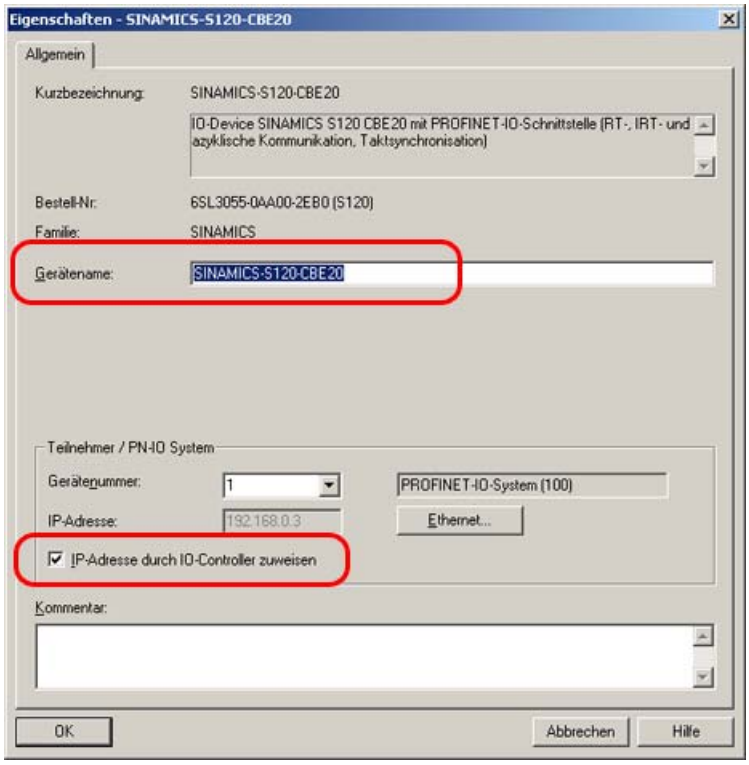


Bild 4-31 Eigenschaften SINAMICS S120

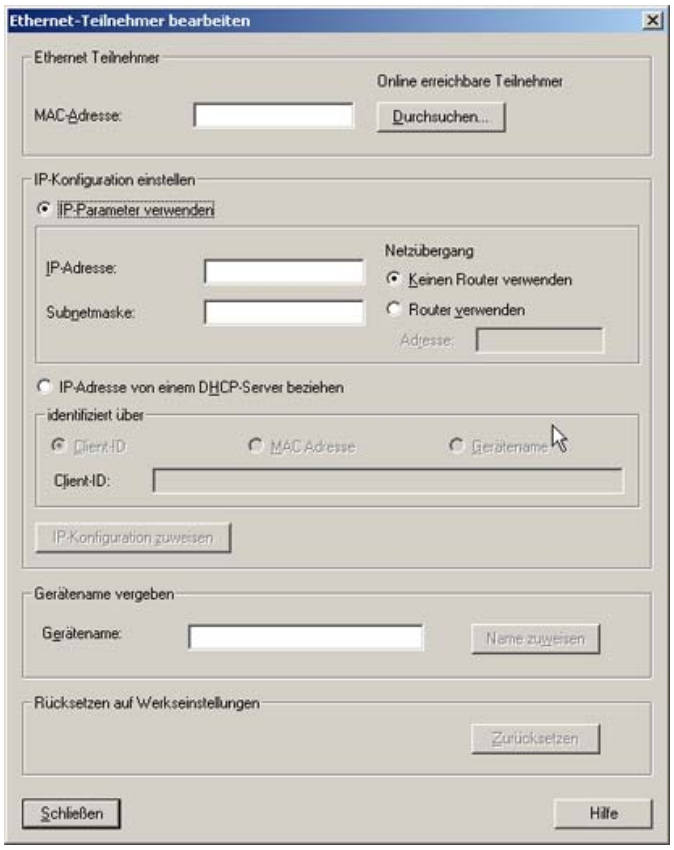
Taufe eines IO-Devices

Hinweis

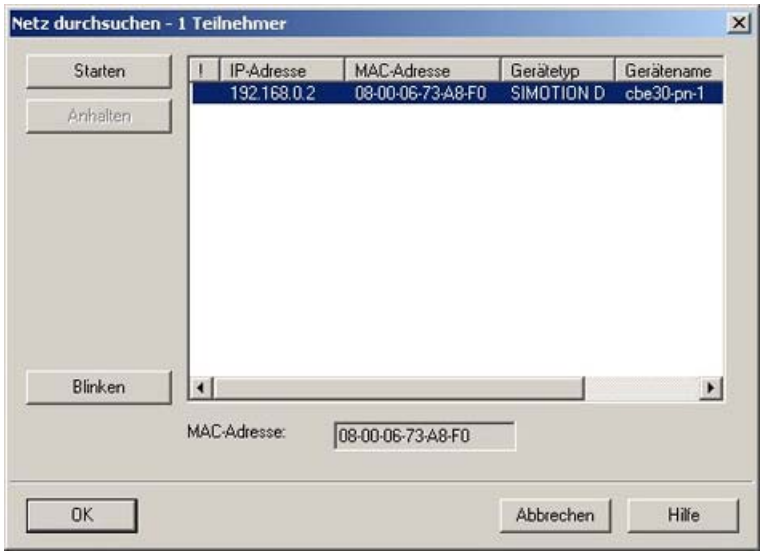
Wenn Sie das PG/PC direkt an der PROFINET-Schnittstelle des Gerätes anschließen, können Sie ein Patch oder Crossover-Kabel verwenden.

Bei der Inbetriebnahme wird empfohlen, sich direkt mit dem zu taufenden Gerät mit dem PG/PC zu verbinden.

1. Wählen Sie in HW Konfig oder NetPro den Menüpunkt **Zielsystem - Ethernet - Ethernet-Teilnehmer bearbeiten**. Es öffnet sich der Dialog **Ethernet-Teilnehmer bearbeiten**.



2. Klicken Sie auf den Button **Durchsuchen**.
3. Es öffnet sich der Dialog **Netz durchsuchen**. Die angeschlossenen Teilnehmer werden angezeigt.



4. Klicken Sie das zu taufende Gerät an und bestätigen Sie mit **OK**.

5. Geben Sie die IP-Adresse und Subnet Maske, die Sie im Dialog **Eigenschaften - Ethernet Schnittstelle …** festgelegt haben, an.
6. Die Default-Einstellung (**Keinen Router verwenden**) für den Netzübergang bleibt unverändert.
7. Klicken Sie den Button **IP-Konfiguration zuweisen**. Die IP-Adresse wird dann online dem Gerät zugewiesen
8. Geben Sie den Gerätenamen ein, den Sie in HW Konfig festgelegt haben, siehe Bild **Eigenschaften SINAMICS S120**.
9. Klicken Sie den Button **Name** zuweisen. Es wird der Gerätename dem Gerät zugewiesen.

Alternativ Knotentaufe in SIMOTION SCOUT durchführen

Sie können die Knotentaufe auch im SCOUT durchführen.

- Führen Sie im SCOUT **Erreichbare Teilnehmer** aus und klicken Sie im eingeblendeten Dialog mit der rechten Maustaste das Gerät an, das Sie bearbeiten möchten.
- Führen Sie **Ethernet Teilnehmer bearbeiten** aus. Der entsprechende Dialog wird eingeblendet.

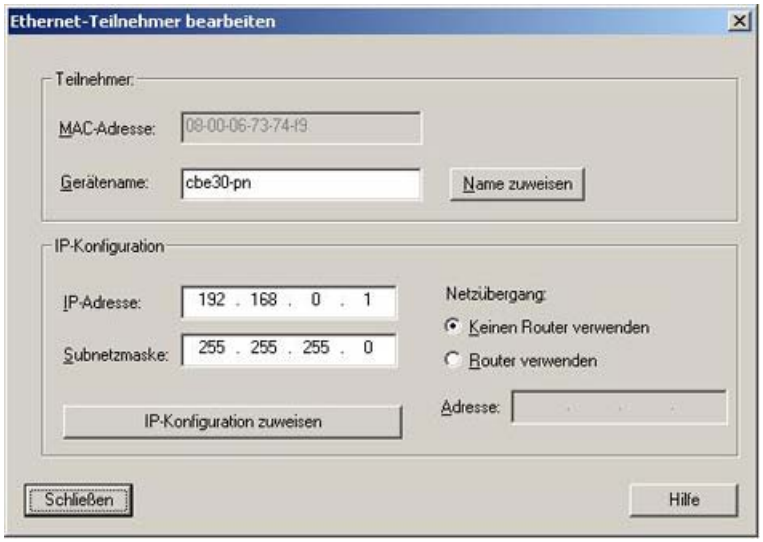


Bild 4-32 Ethernet Teilnehmer bearbeiten

- Geben Sie einen Gerätenamen, eine Subnetzmaske und eine IP-Adresse ein.
- Bestätigen Sie die Eingaben.

Gerätename und IP-Adresse werden in das Gerät übertragen und dort gespeichert.

4.3.14 IP-Adresse und Kommunikationsnamen per AWP/DCP (Mini-IP-Config)

Beschreibung

Bisher unterstützte SIMOTION nur für I-Devices die Vergabe der IP-Adresse und des Gerätenamens (NameOfStation) aus dem Anwenderprogramm (AWP) bzw. über das Discovery Configuration Protocoll (DCP). Ab SIMOTION V4.2 und Step 7 V5.5 muss dies in HW Konfig projektiert werden und gilt für alle IO Devices und IO Controller. Dazu müssen bei den Eigenschaften der Ethernet-Schnittstelle und des PROFINET-Interfaces die Checkboxen für die freie Vergabe gesetzt werden.

Mit diesem Mechanismus können ohne Projektänderung die IP-Adressen und Stationsnamen angepasst werden. Insbesondere für Serienmaschinen können die IP-Einstellungen vor Ort geändert werden.

IP-Adresse auf anderem Weg beziehen

1. Öffnen Sie in HW Konfig den Eigenschaftsdialog des PROFINET-Interfaces und klicken Sie im Register Allgemein auf den Button Eigenschaften.
2. Im aufgeblendeten Eigenschaftsdialog der Ethernet-Schnittstelle wechseln Sie in das Register **Parameter**.
3. Aktivieren Sie die Checkbox **IP-Adressse auf anderem Weg beziehen** und bestätigen Sie mit **OK**.

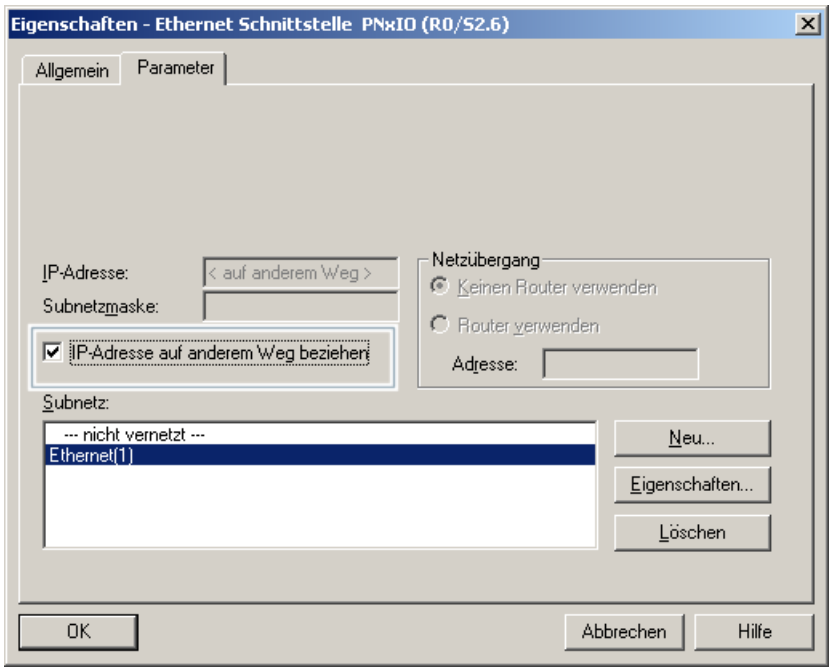


Bild 4-33 IP-Adresse auf anderem Weg beziehen

Gerätenamen auf anderem Weg beziehen

1. Öffnen Sie in HW Konfig den Eigenschaftsdialog des PROFINET-Interfaces.
2. Aktivieren Sie im Register Allgemein die Checkbox **Gerätenamen auf anderem Weg beziehen** und bestätigen Sie mit **OK**.

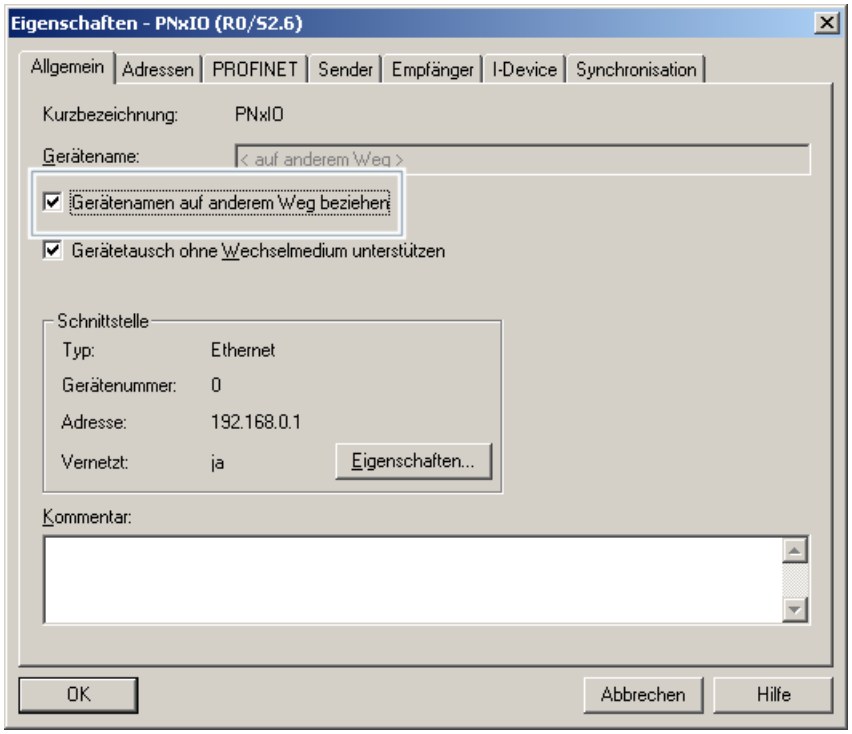


Bild 4-34 Gerätenamen auf anderem Weg beziehen

Allgemeines zur Änderung der IP-Adresse

- Die IP-Adresse kann zu jedem Zeitpunkt geändert werden.
- Ein Hochfahren ohne gültige IP Adresse ist möglich.
- Der Wechsel der IP-Suite (IP-Adresse, Subnet, Router-Adresse, Gerätenamen) inklusive der Übergang auf eine nicht gültige IP-Adresse ist auch im RUN und bei laufender Kommunikation möglich sein. Dabei wird ein Stoß für die Applikation akzeptiert.
- Für den PROFINET IO Controller ist während des Wechsels der IP-Suite ein Geräteausfall und -wiederkehr akzeptiert.
- Eine Änderung der IP-Adresse über DCP bei stehender Applikations-Verbindung (AR - Application Relation) zu einem überlagerten IO Controller wird nicht negativ quittiert. Dieses Verhalten gilt auch bei Nutzung als I-Device.
- Bei Änderung über das AWP wird die AR zum überlagerten IO Controller lokal abgebrochen und der Wechsel akzeptiert. Dies wirkt sich im überlagerten IO Controller als Stationsausfall aus.

Allgemeines zur Änderung des Gerätenamens

- Der Gerätename kann zu jedem Zeitpunkt geändert werden.
- Ein Hochfahren ohne gültigen Gerätenamen ist möglich.
- Der Wechsel der IP-Suite (IP-Adresse, Subnet, Router-Adresse, Gerätenamen) inklusive der Übergang auf eine nicht gültige IP-Adresse ist auch im RUN und bei laufender Kommunikation möglich sein. Dabei wird ein Stoß für die Applikation akzeptiert.
- Der Wechsel des Gerätenamens inklusive der Übergang auf einen "leeren" Gerätenamen ist im RUN und bei laufender Kommunikation möglich sein. Dabei wird ein Stoß für die Applikation akzeptiert.
- Eine Änderung des Gerätenamens über DCP mit stehender Applikations-Verbindung führt zum Abbruch der Applikations-Verbindung. Dies gilt auch bei Nutzung als I-Device.
- Bei einem IO Controller wird beim Wechsel des Gerätenamens alle bestehenden Applikations-Verbindung zu unterlagerten IO Devices abgebrochen und erneut aufgebaut.
- Der eindeutige Gerätename wird nicht überprüft, eine Duplizität kann nur am überlagerten IO Controller bzw. in der Lifelist erkannt werden.

Diagnose im Fehlerfall

Zur Fehlerdiagnose wird ein Diagnosepuffereintrag an der SIMOTION I-Device CPU erzeugt, wenn ein DCP-Auftrag zum Setzen des Gerätenamens oder der IP-Adresse für das I-Device nicht ausgeführt werden konnte, weil in HW Konfig die entsprechenden Checkboxen nicht aktiviert sind.

4.4 Direkten Datenaustausch zwischen IO Controllern projektieren

4.4.1 Einleitung

Zwischen zwei oder mehreren SIMOTION-Controllern können E/A-Datenbereiche zyklisch über IRT Hohe Performance ausgetauscht werden. Dies wird auch als Controller/Controller-Querverkehr bezeichnet. Der Controller/Controller-Querverkehr ist nur über PROFINET IO mit IRT Hohe Performance zwischen SIMOTION-Controllern möglich.

Für den Datenaustausch müssen sich die Geräte in einer gemeinsamen Sync-Domain befinden und entsprechend als Sync-Master und Sync-Slave konfiguriert sein.

Hinweis

Für SIMATIC CPUs steht diese Funktion nicht zur Verfügung.

Es gibt zwei Arten von Querverkehr, den vom System automatisch angelegten Querverkehr z. B. verteilter Gleichlauf und den applikativen, vom Anwender in seiner Applikation nutzbaren Querverkehr. Diesen Querverkehr können Sie projektieren.

Hinweis

Der vom System automatisch projektierte Querverkehr darf nicht durch den Anwender in den Engineering Tools verändert werden (z. B. Veränderung der Adressbereiche). Dies führt zu Fehlerzuständen!

Empfehlung

Wir empfehlen, zunächst die Sendebereiche für alle PROFINET-Geräte zu projektieren und anschließend die Empfangsbereiche. Bei dieser Vorgehensweise können Sie bei der Definition der Empfangsbereiche die zuvor definierten Sendebereiche zuweisen. Damit werden Fehleingaben vermieden.

Datenmenge

Es können ca. 3 KByte übertragen werden. Für jede projektierte Gleichlaufbeziehung werden 24 Byte benötigt. D. h., wurden zu einer Leitachse 5 Folgeachsen definiert, benötigt das System 5 * 24 Byte. Die verbleibende Datenmenge steht für den applikativen Querverkehr zur Verfügung.

Hinweis

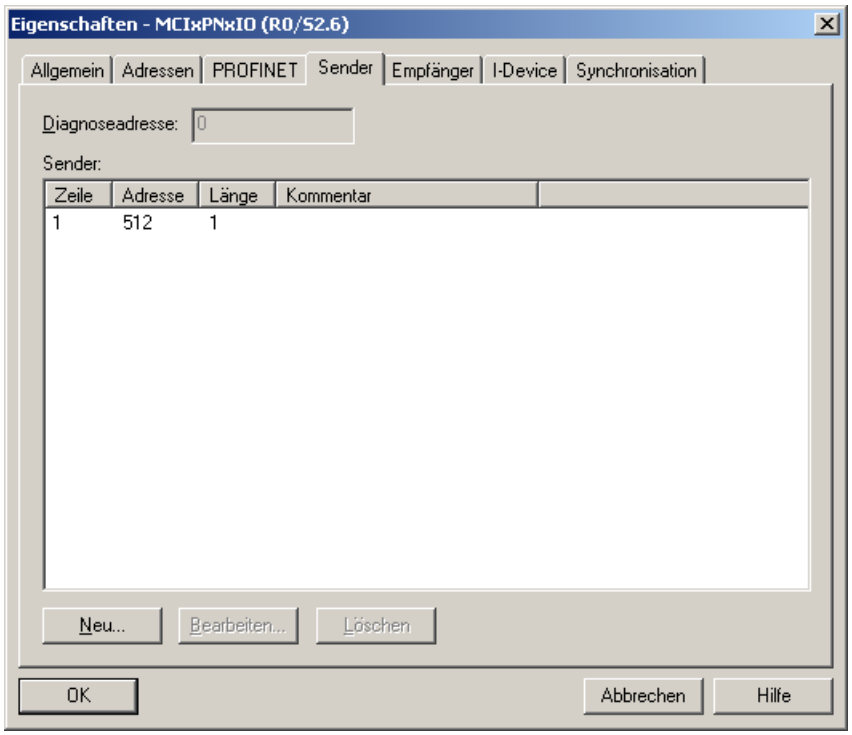
In den SIMOTION Utilities & Applications ist ein FAQ zum Thema PROFINET Projektierung beschrieben. Die SIMOTION Utilities & Applications sind im Lieferumfang von SIMOTION SCOUT enthalten.

In diesem FAQ werden die Themen Verteilter Getriebegleichlauf und Controller/Controller-Querverkehr behandelt.

4.4.2 Sender projektieren

Vorgehensweise

1. Öffnen Sie den Eigenschaftsdialog des PROFINET Interfaces (Doppelklick auf die entsprechende Zeile in der Konfigurationstabelle von HW Konfig).
2. Wählen Sie das Register **Sender**.

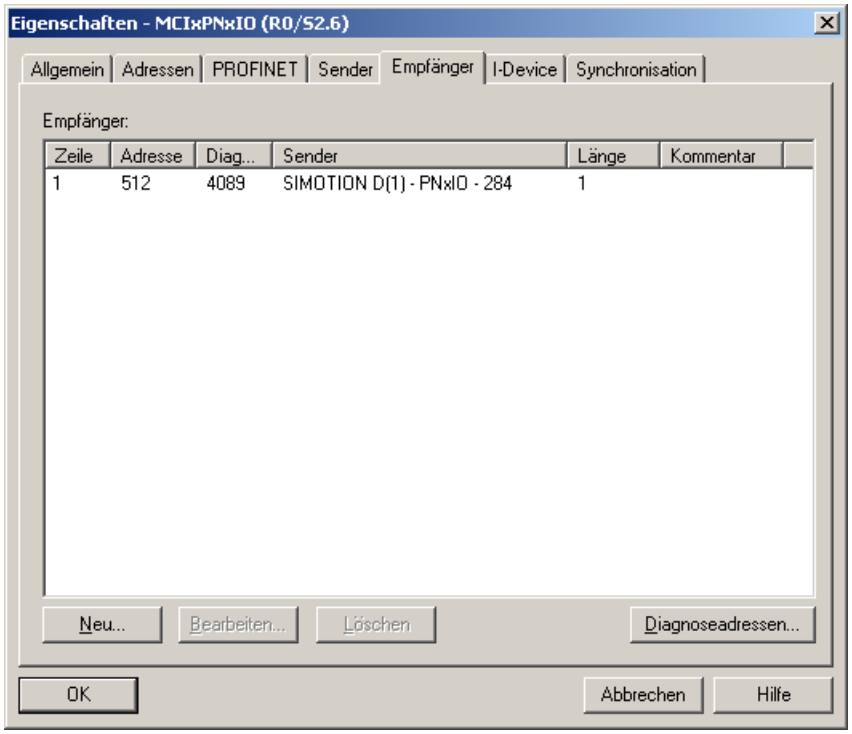


3. Klicken Sie auf die Schaltfläche **Neu**.
4. Geben Sie im Eigenschaftsdialog des Senders Anfangsadresse aus dem I/O-Bereich und Länge des Adressbereichs ein, über den gesendet werden soll. Kommentieren Sie den Datenbereich, um später die über diesen Bereich gesendeten Daten identifizieren zu können. Die maximale Größe einer Variablen ist auf 254 Byte begrenzt.
5. Quittieren Sie die Einstellungen mit **OK**.
6. Wiederholen Sie die Schritte 3 bis 5 für weitere Sendebereiche.
7. Ändern Sie, wenn gewünscht, die voreingestellte Diagnoseadresse für die Sendebereiche.
8. Bestätigen Sie Ihre Eingabe mit **OK**.

Für die Kommunikationsbeziehung, in der ein PROFINET-Interface als Sender für direkten Datenaustausch auftritt, ist genau eine Diagnoseadresse zu vergeben.

4.4.3 Empfänger projektieren

Vorgehensweise



1. Öffnen Sie den Eigenschaftsdialog des PROFINET Interfaces (Doppelklick auf die entsprechende Zeile in der Konfigurationstabelle von HW Konfig).
2. Wählen Sie das Register **Empfänger**.
3. Klicken Sie auf die Schaltfläche **Neu**.
4. Klicken Sie im Dialog **Eigenschaften Empfänger** auf die Schaltfläche **Sender** zuordnen.
5. Wählen Sie im Dialog **Sender zuordnen** den Datenbereich der gewünschten Station aus, der vom lokalen Controller empfangen werden soll.
6. Quittieren Sie die Auswahl mit **OK**.
7. Geben Sie im Eigenschaftsdialog des Empfängers Anfangsadresse des Adressbereichs ein, über den empfangen werden soll. Die Länge des Adressbereichs darf nicht geändert werden, da sie automatisch an die Länge des Sendebereichs angepasst wird. Nur wenn Sende- und Empfangsbereiche identische Längen haben, ist die Konfiguration übersetzbar!
8. Wiederholen Sie die Schritte 3 bis 7 für weitere Empfangsbereiche.
9. Für jeden zugeordneten Sender ist eine Diagnoseadresse reserviert, über die der Empfänger einen Ausfall des Senders feststellen kann.
10. Kicken Sie auf die Schaltfläche **Diagnoseadressen**, wenn Sie diese Adressen editieren wollen.
11. Bestätigen Sie Ihre Eingabe mit **OK**.

4.5 I-Device projektieren

4.5.1 PROFINET IO und I-Device

Einführung

Die direkte Kopplung z. B. von SIMATIC und SIMOTION über PROFINET war bis SIMOTION 4.0 nur mittels TCP oder UDP bzw. zusätzlicher Hardware (PN/PN-Coupler, SIMATIC-CP) möglich. Ab SIMOTION V 4.1.1.6 wurde für PROFINET IO die von PROFIBUS bekannte direkte Kopplung der Steuerungen realisiert. Über PROFIBUS kann z. B. die SIMOTION als I-Slave an die SIMATIC CPU angebunden werden. Eine vergleichbare Funktion steht auch für PROFINET IO unter der Bezeichnung I-Device zur Verfügung. Damit ist der Datenaustausch zwischen den Steuerungen über E/A-Bereiche möglich. Dadurch wird die für TCP oder UDP notwendige Programmierung der Kommunikation durch Projektierung und Systemfunktionalität ersetzt. Weiterhin entfallen die Kosten für bisher eingesetzten Hardware-Lösungen (PN/PN-Coupler, SIMATIC-CP).

Ein I-Device ist ein Controller, der zusätzlich die Funktion eines IO-Devices übernimmt. Der Begriff I-Device steht für Intelligentes IO-Device. Merkmal eines Intelligenten I-Devices ist, dass dessen Ein-/Ausgangsdaten nicht unmittelbar von realen Ein-/Ausgängen dem übergeordneten IO-Controller zur Verfügung gestellt werden, sondern dass eine Vorverarbeitung der Daten im I-Device stattfindet.

Ein SIMOTION-Gerät als I-Device kann z. B. zum Datenaustausch zu einer SIMATIC-Station verwendet werden. Darüber hinaus kann beispielsweise ein SIMOTION-Gerät als I-Device auch als Anleger einer modularen Maschine verwendet werden. Siehe dazu *Funktionsbeschreibung Motion Control Basisfunktionen für modulare Maschinen*.

Weiterhin kann ein SIMOTION-Gerät als I-Device auch für einen verteilten Gleichlauf über Projektgrenzen hinweg eingesetzt werden, siehe dazu Funktionshandbuch Motion Control Technologieobjekte Gleichlauf, Kurvenscheibe.

Hinweis

Ein I-Device kann erst ab SIMOTION V4.1.1.6 angelegt werden.

Eigenschaften eines I-Devices

Ein I-Device kann neben der Rolle eines IO-Devices an einem übergeordneten IO-Controller gleichzeitig auch ein eigenes lokales PROFINET IO-System aufspannen mit eigenen, lokalen IO-Devices, somit selbst auch ein IO-Controller sein. Beide Funktionen werden über ein und dieselbe PROFINET-Schnittstelle des Gerätes realisiert.

Das I-Device steht bei SIMOTION für PROFINET IO mit RT und mit IRT Hohe Performance zur Verfügung.

Für die Kombinationsmöglichkeit der Funktionen gibt es folgende Randbedingung:

Tabelle 4- 4 Kombinationsmöglichkeiten RT und IRT I-Device bei SIMOTION

Funktion der SIMOTION	Welche zusätzlichen Funktionen sind noch möglich			
	RT I-Device	RT-Controller	IRT I-Device	IRT-Controller
RT I-Device		X	-	X
RT-Controller	X	-	X*	X*
IRT I-Device	-	X	-	-
IRT-Controller	X	X	-	

*entweder IRT I-Device oder IRT-Controller

Die PROFINET-Schnittstelle eines I-Devices benötigt wie jedes andere IO-Device zum Betrieb Parametrierdaten. Bei einem IO-Device werden diese generell über den zugehörigen IO-Controller in Form von Parametrierdatensätzen geladen. Bei einem I-Device stehen 2 Möglichkeiten zur Verfügung. Das Interface und die Ports der PROFINET-Schnittstelle eines I-Devices können entweder vom übergeordneten IO-Controller oder lokal durch das I-Device selbst parametriert werden. Dies kann in der Konfiguration des I-Devices ausgewählt werden.

Bei der lokalen Parametrierung werden die notwendigen Daten beim Download vom Engineering-System in das I-Device geladen. Die Parametrierdaten für die PROFINET-Schnittstelle sind in den Downloaddaten für das Gerät enthalten. Der übergeordnete IO-Controller muss die PROFINET-Schnittstelle des I-Devices nicht parametrieren. Diese Möglichkeit ist zu verwenden, wenn das I-Device mit RT betrieben werden soll.

Bei der Parametrierung durch den übergeordneten IO-Controller müssen die Parametrierdaten für die PROFINET-Schnittstelle des I-Devices zusammen mit den übrigen Parametrierdaten durch den IO-Controller geladen werden. Der IO-Controller lädt dazu Parametrierdatensätze für die PROFINET-Schnittstelle in das I-Device. Wenn das I-Device mit IRT betrieben werden soll, dann müssen die Parametrierdaten durch den IO-Controller geladen werden.

Wenn das I-Device mit IRT betrieben wird, dann muss der Sendetakt des I-Devices gleich dem Sendetakt der Sync-Domain des PROFINET IO-Systems des übergeordneten IO-Controllers eingestellt werden. Wenn das I-Device mit RT betrieben wird, dann muss die Aktualisierungszeit des I-Devices gleich oder um ein Vielfaches untersetzt zum Sendetakt der Sync-Domain des PROFINET IO-Systems des übergeordneten IO-Controllers eingestellt werden.

Folgende Sendetakte und Aktualisierungszeiten in den folgenden möglichen Kombinationen müssen, wie in der Tabelle unten angegeben, eingestellt werden.

Tabelle 4- 5 Sendetakte/Aktualisierungszeiten eines I-Devices

Übergeordneter IO-Controller und I-Device mit IRT, kein lokales PROFINET IO-System oder lokales PROFINET IO-System mit IO-Devices mit RT • Sendetakt I-Device: – muss gleich zum Sendetakt des übergeordneten IO-Controllers sein. – einzustellen am I-Device in Eigenschaften < PROFINET Interface> unter dem Reiter PROFINET in der Drop-Down Box "Sendetakt"
Übergeordneter IO-Controller mit IRT und I-Device mit RT, lokales PROFINET IO-System mit IRT • Aktualisierungszeit I-Device: – muss ein ganzzahliges Vielfaches des Sendetaktes des übergeordneten IO-Controllers und des Sendetaktes des IO-Controllers im I-Device sein – einzustellen am I-Device-Stellvertreter in Eigenschaften <Profinet Interface> im Reiter IO-Zyklus unter Aktualisierungszeit
Übergeordneter IO-Controller und I-Device mit RT, kein lokales PROFINET IO-System • Aktualisierungszeit I-Device: – Es können die für das I-Device möglichen Aktualisierungszeiten eingestellt werden – einzustellen am I-Device-Stellvertreter in Eigenschaften <PROFINET Interface> im Reiter IO-Zyklus unter Aktualisierungszeit
Übergeordneter IO-Controller und I-Device mit RT, lokales PROFINET IO-System mit IRT: • Aktualisierungszeit I-Device: – muss gleich oder kleiner als der Sendetakt des IO-Controllers im I-Device sein. – einzustellen am I-Device-Stellvertreter in Eigenschaften <PROFNET Interface> im Reiter IO-Zyklus unter Aktualisierungszeit

Das folgende Bild zeigt, wie ein I-Device an einem übergeordneten IO-Controller projektiert werden kann. Der übergeordnete IO-Controller spannt ein PROFINET IO-System auf, in dem sich das I-Device befindet. Das I-Device kann ein lokales PROFINET IO-System aufspannen. Jedes dieser PROFINET IO-Systeme kann einer eigenen Sync-Domain angehören. Das I-Device darf aber nur einer der möglichen Sync-Domains zugeteilt werden, da eine PROFINET-Schnittstelle nur genau einer Sync-Domain angehören kann.

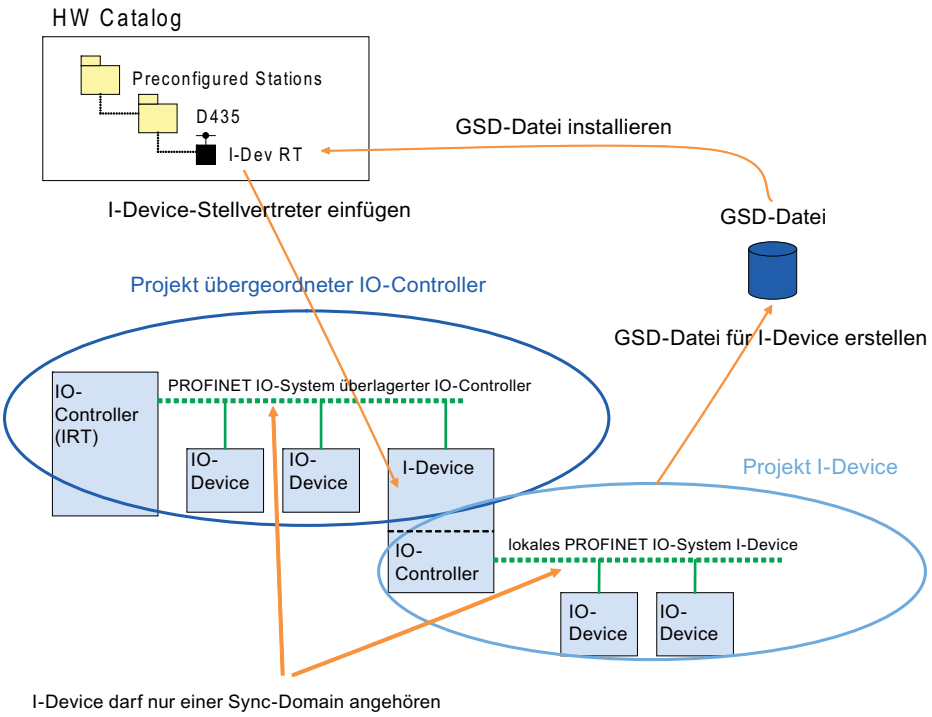


Bild 4-35 Projektierung I-Device

Vorgehen beim Projektieren

- Das I-Device selbst und der IO-Controller, an dem es betrieben werden soll, sollte in verschiedenen Projekten angelegt werden.
- Für das I-Device muss der I-Device Modus der PROFINET-Schnittstelle eingeschaltet werden. Weiterhin müssen die Ein- und Ausgangsbereiche im I-Device für den Datenaustausch mit dem übergeordneten IO-Controller konfiguriert werden.
- Nach dem Anlegen und Projektieren eines I-Devices muss eine GSD-Datei für dessen I-Device-Stellvertreter erstellt und installiert werden. Der I-Device-Stellvertreter steht danach im Hardware Katalog unter Preconfigured Stations zur Verfügung.
- Im Anschluss muss der I-Device-Stellvertreter aus dem Hardware Katalog aus Preconfigured Stations in das PROFINET IO-System des übergeordneten IO-Controllers eingefügt werden.

Da ein I-Device-Stellvertreter nur über einen manuellen Vorgang im Hardware Katalog erzeugt werden kann, findet damit kein automatischer Abgleich zwischen dem Projekt mit dem I-Device und dem entsprechenden I-Device-Stellvertreter in der GSD-Datei statt. Folglich darf die Konfiguration des I-Devices nachträglich nicht mehr verändert werden. Wenn jedoch eine Änderung stattfindet, dann bedingt dies eine neue GSD-Datei, die wieder erzeugt und installiert werden muss. Wenn die Konfiguration eines bestimmten I-Devices mehrmals nachträglich geändert und mehrmals von diesem I-Device eine GSD-Datei erstellt und installiert wird, erscheint immer die neueste Version im Hardware Katalog unter Preconfigured Stations. Um das Erneuern der Version sicherzustellen, muss jedoch der beim Erstellen und Installieren der GSD-Datei verwendete Bezeichner für den I-Device-Stellvertreter identisch sein. Nur die Ein- bzw. Ausgangsadressen für den Datenaustausch dürfen im Projekt des übergeordneten IO-Controllers geändert werden.

Da I-Devices an ihren übergeordneten IO-Controller und die am PROFINET IO-System eines einzelnen I-Devices angeschlossenen IO-Devices über ein und dieselbe PROFINET-Schnittstelle angeschlossen werden, befinden sich alle die genannten Geräte auch in ein und demselben Ethernet-Subnetz. Dies hat zur Konsequenz, dass die Gerätenamen und IP-Adressen aller dieser Geräte voneinander verschieden bzw. die Subnetzmasken identisch sein müssen. Diesem Fakt ist dann besondere Aufmerksamkeit zu schenken, wenn sich der übergeordnete IO-Controller und das I-Device in verschiedenen Projekten befinden. Denn HW Konfig kann die Konsistenz der Gerätenamen, IP-Adressen und Subnetzmasken über verschiedene Projekte hinweg nicht prüfen.

Gerätename (NameOfStation) für I-Device

Wie für alle IO-Devices an PROFINET IO muss auch für ein I-Device ein Gerätenamen in der Projektierung festgelegt werden. Der Gerätename (NameOfStation) für das I-Device wird in den Eigenschaften von dessen PROFINET-Schnittstelle eingestellt und ist damit identisch mit dem Gerätenamen des IO-Controllers im I-Device. Dieser eingestellte Name wird beim Erstellen und Installieren der GSD-Datei für den I-Device-Stellvertreter mit in die GSD-Datei geschrieben. Beim Einfügen des I-Device-Stellvertreters in das PROFINET IO-System des übergeordneten IO-Controllers wird der in der GSD-Datei vorbelegte Gerätename übernommen in die Konfiguration. Auf jeden Fall muss sichergestellt werden, dass der Gerätename in der Konfiguration des übergeordneten IO-Controllers identisch ist zum Gerätenamen, der für das I-Device festgelegt ist. Folglich darf der Gerätenamen nach dem Hinzufügen in das PROFINET IO-System des übergeordneten IO-Controllers nicht mehr geändert werden.

Wenn die Gerätenamen verschieden sind, dann kann der übergeordnete IO-Controller das betreffende I-Device nicht hochfahren und infolgedessen nicht den zyklischen Austausch der Ein-/Ausgangsdaten starten.

Folgende Fälle führen zu verschiedenen Gerätenamen und müssen deshalb vermieden werden:

- Wenn Sie als übergeordneten IO-Controller ein SIMOTION-Gerät verwenden, darf der Gerätename (NameOfStation) des I-Device keine "-" enthalten. Denn diese werden in "x" umgewandelt, wenn das I-Device in das PROFINET IO-System eingefügt wird.
- Da ein Gerätename innerhalb eines Ethernet-Subnetzes nicht zweimal vorkommen darf, wird beim Einfügen eines I-Device-Stellvertreters in das PROFINET IO-System seines übergeordneten IO-Controllers der Gerätename geändert, wenn dieser schon vorhanden ist. Aus diesem Grund muss dafür gesorgt werden, dass der in der GSD-Datei vorbelegte Gerätenamen noch nicht verwendet wird.
- Wenn mehr als ein I-Device-Stellvertreter aus ein und demselben Eintrag von Preconfigured Stations in das PROFINET IO-System des übergeordneten IO-Controllers eingefügt wird, dann wird der in der GSD-Datei vorbelegte Gerätename verändert. Deshalb muss für jedes I-Device, welches in einem PROFINET IO-System verwendet werden soll, auch je ein I-Device-Stellvertreter angelegt werden.

Siehe auch

I-Device anlegen (Seite 125)

4.5.2 I-Device Funktionalität ab SIMOTION SCOUT V4.2

Beschreibung

Ab Step 7 5.5 können auch SIMATIC CPUs als I-Device projektiert werden. Die I-Device Funktionalität von SIMOTION CPUs und SIMATIC CPUs wurde durchgängig vereinheitlicht. Im Rahmen der Vereinheitlichung wurde die GSD Version auf V2.25 gesetzt. Bei SIMOTION-Projekten < V4.2 muss daher beim Hochrüsten auf V4.2 das I-Device neu exportiert/ installiert und im Projekt des neu überlagerten Controllers eingebunden werden. Bearbeiten Sie Projekte ohne die GSD-Datei erneut zu Exportieren und zu Installieren, wird beim Verbindungsaufbau über PROFINET ein Diagnosepuffereintrag an der CPU des I-Devices erzeugt.

Hinweis

Randbedingungen I-Device ab V4.2

Für die Projektierung am überlagerten Controller muss Step 7 V5.5 verwendet, da nur ab dieser Version die GSD V2.25 importiert werden kann.

Beim Hochrüsten von älteren Versionen der SIMOTION CPU auf V4.2 wird, wird das I-Device "inkompatibel" und ein Ex-/Import der GSD muss auf jedem Fall durchgeführt werden.

Im Folgenden finden Sie verschiedene Szenarien für eine Kompatibilität des I-Devices aufgeführt.

UseCase 1: Altes Projekt mit SIMOTION Geräten < V4.2 ohne Änderung am I-Device.

1. Sie öffnen ein altes Projekt mit SCOUT V4.2/Step 7 5.5. Im Projekt sind RT I-Device für die Kommunikation zu einer SIMATIC CPU enthalten.
2. Sie ändern das Projekt, aber Sie nehmen **keine** Änderungen an der I-Device Konfiguration vor.
3. Sie übersetzen das SIMOTION Projekt inklusive HW Konfig und speichern es als SIMOTION SCOUT V4.1.
4. Das Projekt kann ohne Probleme wieder geladen werden.

UseCase 2: Altes Projekt mit SIMOTION Geräten < V4.2 mit Änderung an I-Device Schnittstelle.

1. Sie öffnen ein altes Projekt mit SCOUT V4.2/Step 7 5.5. Im Projekt sind RT I-Device für die Kommunikation zu einer SIMATIC CPU enthalten.
2. Sie ändern das Projekt und nehmen Änderungen an der I-Device Konfiguration vor. Es werden automatisch an der I-Device-Schnittstelle Änderungen durchgeführt und ein IO-Slot kommt dazu.
3. Erzeugen Sie eine neue GSD V2.25 vom I-Device.
4. Installieren Sie das exportierte I-Device und ersetzen es an der übergeordneten SIMATIC CPU.

   Hinweis

   Verwenden Sie die neue GSD in einem anderen Projekt, so kann dies nur noch mit STEP 7 V5.5 bearbeitet werden, da nur damit die Kompatibilität mit GSD V2.25 sicher gestellt ist.

5. Sie übersetzen das SIMOTION Projekt inklusive HW Konfig und speichern es als SIMOTION SCOUT V4.1.
6. Das Projekt kann ohne Probleme wieder geladen werden.

UseCase 3: Überlagerte SIMATIC CPU und SIMOTION I-Device < V4.2 und Hochrüstung auf V4.2.

1. Sie öffnen das Projekt mit einer SIMOTION CPU und überlagerter SIMATIC CPU. Die SIMATIC CPU kommuniziert als PN-Controller mit dem I-Device der SIMOTION CPU.
2. Sie rüsten die SIMOTION CPU auf SIMOTION V4.2 hoch.
3. Exportieren Sie das I-Device der SIMOTION CPU.
4. Diagnoseadressen und Stationsnummer haben sich gegenüber dem Vorgänger I-Device geändert. Werden diese als absolute Werte in der SIMOTION Applikation in Systemaufrufen verwendet, so muss das Applikationsprogramm anpasst werden.
5. Löschen Sie das bisherige I-Device der SIMOTION CPU im Projekt der überlagerten SIMATIC CPU. Stellen Sie sicher, dass die In- und Outputadressen des neuen I-Devices identisch zu den alten I-Device sind, damit die bisherige S7 Applikation verwendet werden kann.

6. Importieren Sie die neue GSD-Datei in das Projekt der überlagerten SIMATIC CPU.
7. Sie übersetzen das SIMOTION Projekt inklusive HW Konfig und speichern es als SIMOTION SCOUT V4.2. Das Projekt ist auf V4.2 hochgerüstet.

UseCase 4: Überlagerte SIMOTION CPU und mehrere SIMOTION I-Device < V4.2 und Hochrüstung auf V4.2.

1. Sie öffnen das Projekt mit mehreren SIMOTION CPU als I-Device und überlagerter SIMOTION CPU. Die überlagerte SIMOTION CPU kommuniziert als PN-Controller mit dem I-Device der SIMOTION CPU.
2. Sie rüsten die SIMOTION CPUs auf SIMOTION V4.2 hoch.
3. Diagnoseadressen und Stationsnummer haben sich gegenüber dem Vorgänger I-Device geändert. Werden diese als absolute Werte in der SIMOTION Applikation in Systemaufrufen verwendet, so muss das Applikationsprogramm anpasst werden.
4. Exportieren Sie das I-Device der SIMOTION CPUs und importieren die GSD-Datei in das Projekt der überlagerten SIMOTION CPU.
5. Sie übersetzen das SIMOTION Projekt inklusive HW Konfig und speichern es als SIMOTION SCOUT V4.2. Das Projekt ist auf V4.2 hochgerüstet.

4.5.3 I-Device anlegen

Voraussetzung

Sie haben in HW Konfig (SIMATIC Manager oder SIMOTION SCOUT) bereits ein Projekt angelegt und eine Station mit Rack bzw. einen SIMOTION-Controller angelegt. Sie haben das PROFINET IO-System projektiert und möchten nun das I-Device projektieren.

Hinweis

Beachten Sie bei der Projektierung des I-Device die möglichen Einstellungen zur RT-Klasse, siehe dazu PROFINET IO und I-Device (Seite 118).

Vorgehensweise

1. Doppelklicken Sie auf das Interface-Modul der CPU. Der Dialog **Eigenschaften** öffnet sich.
2. Wählen Sie das Register **Allgemein** aus und ändern Sie ggf. den Gerätenamen (keine "-").
3. Wählen Sie das Register **I-Device** aus.

4. Aktivieren Sie die Checkbox **I-Device Modus**.

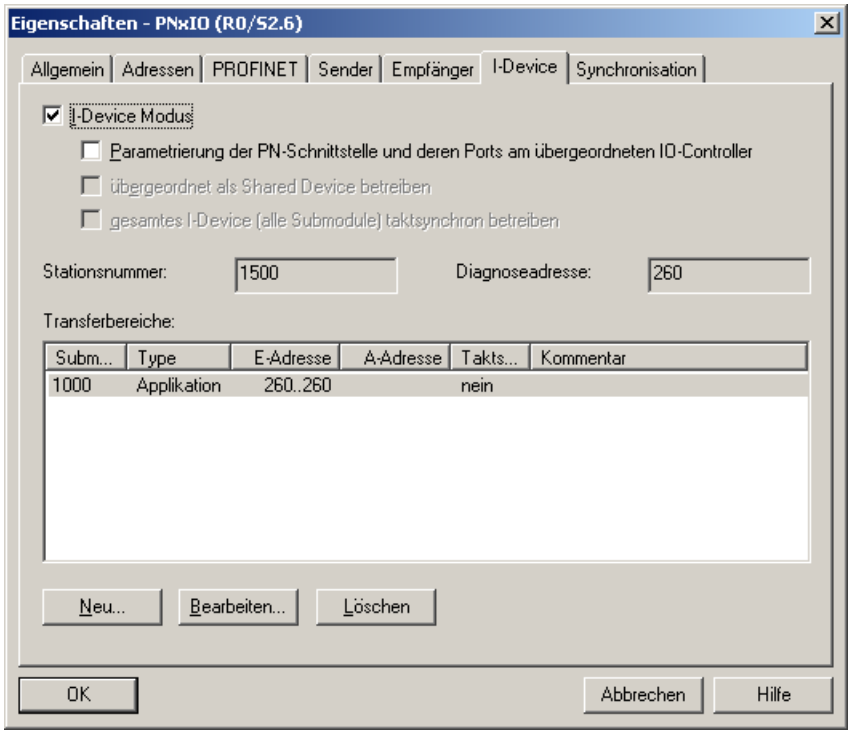


Bild 4-36 I-Device Eigenschaftsdialog

5. Abhängig davon, ob die Kommunikation des I-Device als RT oder IRT am übergeordneten Controller erfolgen soll, müssen Sie unterschiedliche Checkboxen aktiveren.

   – **I-Device als RT:**
   Aktiveren Sie nur die Checkbox **I-Device Modus**.

   – **I-Device als IRT**:
   Wenn das I-Device als IRT am überlagerten IO-Controller betrieben werden soll, müssen Sie die die Checkboxen **Parametrierung der PN-Schnittstelle und deren Ports am überlagerten IO-Controller** und **gesamtes I-Device (alle Submodule) taktsynchron betreiben** zusätzlich aktivieren. Damit werden dann auch Ports in der GSD-Datei angelegt und beim Anlauf werden die Parametrierdatensätze in den Controller des I-Device geladen. Wenn diese Option nicht ausgewählt wird, kann die zyklische Kommunikation zwischen dem übergeordneten IO-Controller und den I-Devices nur über RT erfolgen. Am I-Device-Stellvertreter wird damit im Dialog Eigenschaften der PROFINET-Schnittstelle des I-Device-Stellvertreters das Register **IO-Zyklus** eingeblendet, in dem dann bei **I-Device taktsynchron zuordnen** der Eintrag **Servo** ausgewählt werden kann, um das I-Device taktsynchron betreiben zu können. Wenn ein I-Device mit IRT betrieben werden soll, dann muss sowohl die Parametrierung der PN-Schnittstelle und deren Ports am überlagerten IO-Controller als auch **I-Device taktsynchron zuordnen** gesetzt sein.

Hinweis

Beachten Sie, wenn das I-Device im IRT Betrieb arbeiten soll, sollte der Gerätename und IP-Adresse vom Controller zugewiesen werden. Dazu müssen Sie beim I-Device diese Option setzen. Beachten Sie dazu das Kapitel IP-Adresse und Kommunikationsnamen per AWP/DCP (Mini-IP-Config) (Seite 112) und I-Device-Stellvertreter am übergeordneten IO-Controller einfügen (Seite 130).

6. Wenn das I-Device mit IRT betrieben wird, dann muss dessen Sendetakt eingestellt werden. Wählen Sie dazu das Register **PROFINET** aus und stellen Sie einen entsprechenden Sendetakt ein.
7. Klicken Sie jeweils **Neu**..., um die virtuellen Subslots (Ein- und Ausgangs-Adresse Transferbereich) anzulegen, und konfigurieren Sie diese gemäß den Anforderungen. Damit konfigurieren Sie den E/A-Bereich des I-Device, über den Daten mit dem überlagerten IO-Controller ausgetauscht werden. In den Registern **Sender** und **Empfänger** müssen Sie keine Einstellungen mehr durchführen.
8. Klicken Sie **OK**, um die Einstellungen zu übernehmen und speichern Sie das Projekt.
9. Fahren Sie fort mit I-Device-Stellvertreter erstellen.

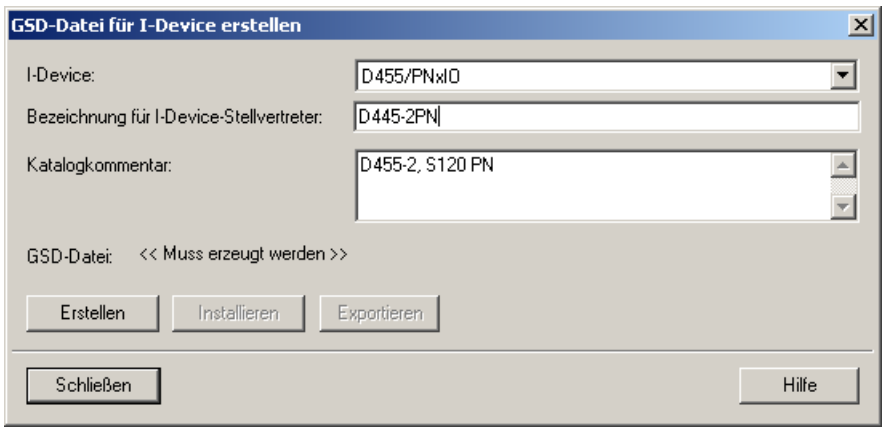
4.5.4 GSD-Datei für I-Device exportieren

Das Exportieren der GSD-Datei wird immer dann benötigt, um ein I-Device in einem Projekt auf einem anderen PC verwenden zu können.

Voraussetzung

Sie haben bereits die Baugruppe, die als I-Device verwendet werden soll, projektiert.

1. Speichern Sie zuerst das Projekt.
2. Führen Sie **Extras > GSD-Datei für I-Device erstellen ...** aus. Der Dialog GSD-Datei für I-Device erstellen wird eingeblendet.
3. Wählen Sie das I-Device aus und geben Sie eine Bezeichnung für den I-Device-Stellvertreter ein. Unter diesem Bezeichner erscheint der I-Device-Stellvertreter unterhalb Preconfigured Stations im HW-Katalog.



4. Klicken Sie auf **Erstellen** und anschließend auf **Exportieren**. Der Dialog **Ordner suchen** wird eingeblendet.
5. Wählen Sie den Pfad aus, in dem die GSD-Datei des I-Device-Stellvertreters abgelegt werden soll, und klicken Sie auf **OK.**

4.5.5 I-Device-Stellvertreter erstellen

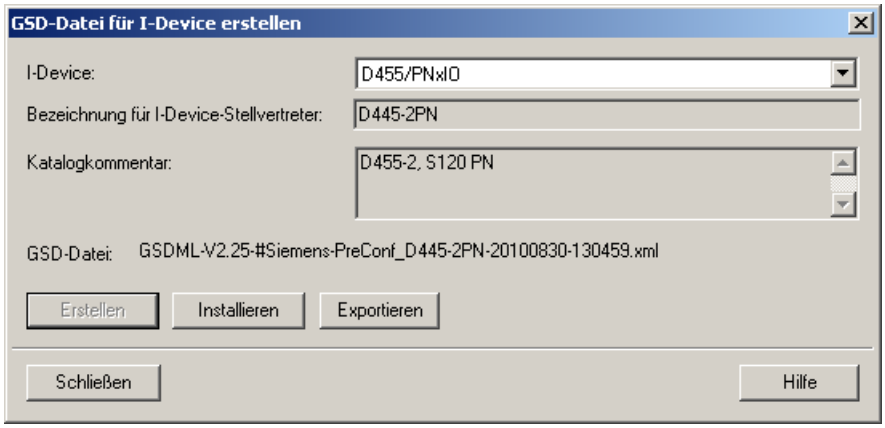
Für das Erstellen eines I-Device-Stellvertreters gibt es 2 Möglichkeiten. Zum einen kann dieser mithilfe einer vorher exportierten GSD-Datei über **Extras > GSD-Dateien installieren ...** und zum anderen im Dialog **GSD-Datei für I-Device erstellen** erzeugt werden.

Voraussetzung

Sie haben bereits die Baugruppe, die als I-Device verwendet werden soll, projektiert und von dieser die GSD-Datei exportiert.

Vorgehensweise 1

1. Speichern Sie zuerst das Projekt.
2. Erstellen Sie wie in Kapitel GSD-Datei für I-Device exportieren (Seite 127) beschrieben ein I-Device. Anstatt es zu exportieren, wird die Datei sofort installiert.
3. Klicken Sie auf **Erstellen** und anschließend auf **Installieren.**

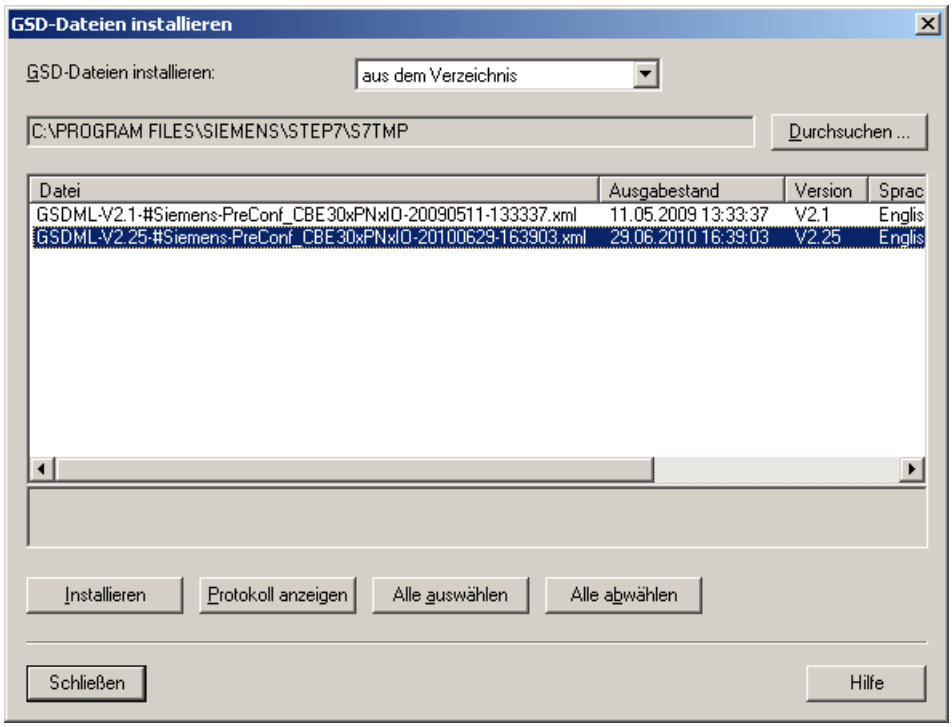


4. Klicken Sie auf **Schließen**. Anschließend steht der I-Device-Stellvertreter unterhalb von Preconfigured Stations zur Verfügung.

Vorgehensweise 2

1. Führen Sie **Extras > GSD-Dateien...** aus. Der Dialog GSD-Dateien installieren wird eingeblendet.
2. Klicken Sie auf **Durchsuchen...** Der Dialog **Ordner suchen** wird eingeblendet.

3. Wählen Sie den Pfad aus, in dem die GSD-Dateien für I-Device-Stellvertreter abgelegt sind, und klicken Sie auf **OK.**



4. Wählen Sie die gewünschten GSD-Dateien aus und klicken Sie auf **Installieren**.
5. Klicken Sie auf **Schließen**. Anschließend stehen die I-Device-Stellvertreter unterhalb von Preconfigured Stations zur Verfügung.

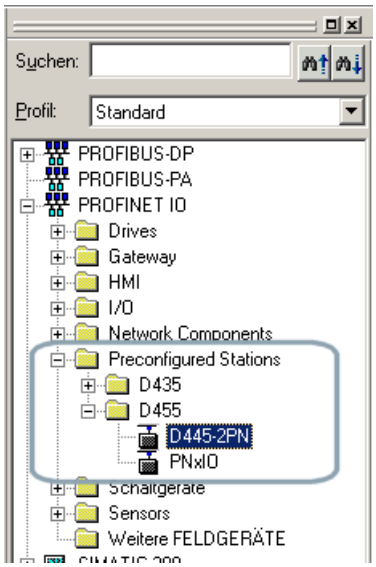


Bild 4-37 i-Device Eintrag im Hardware Katalog

4.5.6 I-Device-Stellvertreter am übergeordneten IO-Controller einfügen

Voraussetzung

Sie haben bereits einen I-Device-Stellvertreter erstellt. Ein Projekt ist geöffnet und ein IO-Controller mit einem PROFINET IO-System ist bereits projektiert.

I-Device-Stellvertreter einfügen

1. Öffnen Sie den Hardware-Katalog.
2. Ziehen Sie den entsprechenden I-Device-Stellvertreter vom Hardware-Katalog (**PROFINET IO > Preconfigured Stations**) auf das PROFINET IO-System. Der I-Device-Stellvertreter wird am PROFINET IO-System wie ein normales IO-Device angezeigt. Abhängig davon, ob **Parametrierung der PN-Schnittstelle und deren Ports am überlagerten IO-Controller** angewählt ist, werden Ports angezeigt oder nicht.

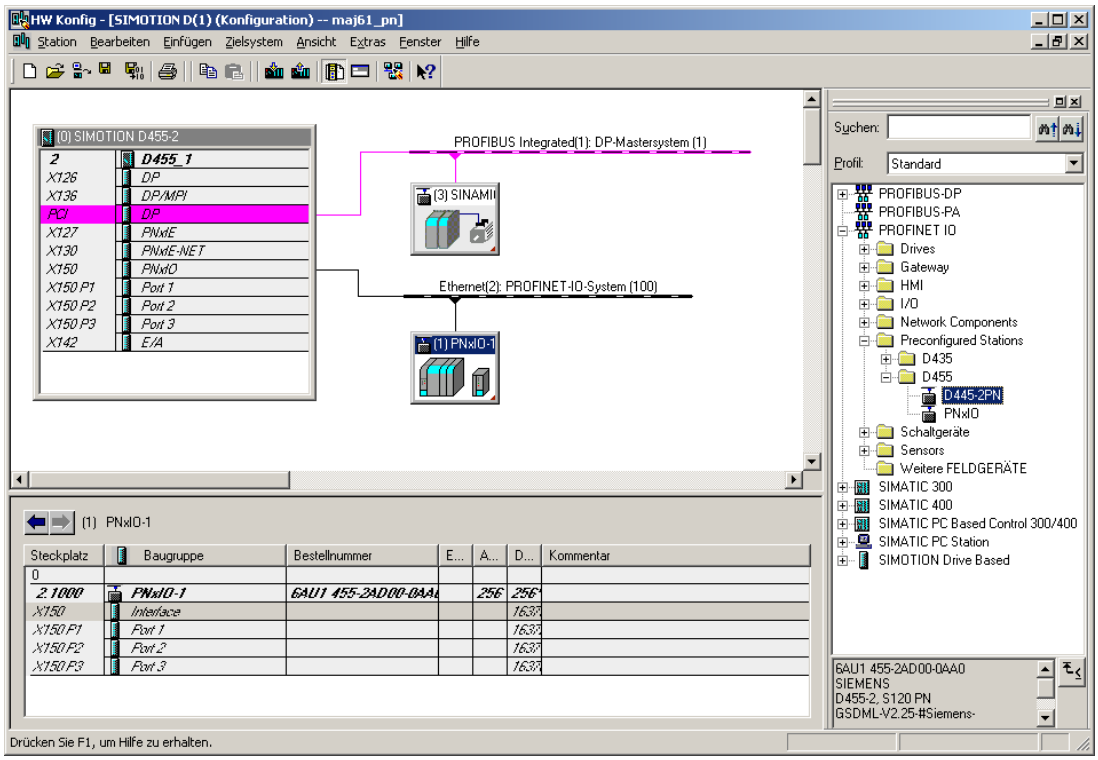


Bild 4-38 I-Device am IO-Controller

Die Anzahl der Submodule entspricht der Anzahl der projektierten Submodule des I-Devices in der GSD-Datei. Modul und Submodule (virtuellen Subslots) sind nicht löschbar.

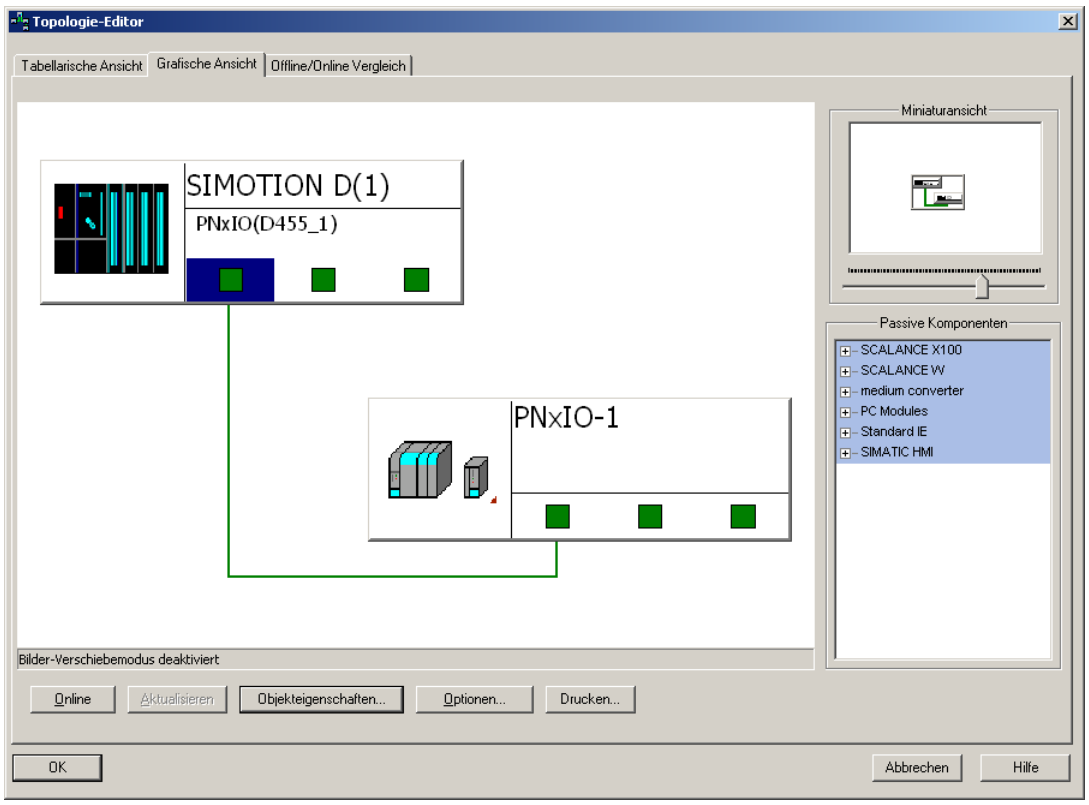


Bild 4-39 I-Device Ports verschalten

IP-Adresse für I-Device-Stellvertreter vergeben

1. Doppelklicken Sie auf das I-Device um den **Eigenschaften** Dialog aufzurufen.
2. Deaktivieren Sie die Option **IP-Adresse durch IO-Controller zuweisen**.

   Die IP-Adresse sollte nicht vom übergeordneten IO-Controller zugewiesen werden, da diese bereits im Step7-Projekt des I-Device vergeben wird.
   Diese Option ist nur verfügbar, wenn Sie vorher bei den Eigenschaften des Geräts **Gerätenamen auf anderem Weg beziehen** und IP-**Adresse auf anderem Weg beziehen** aktiviert haben (siehe auch Mini-IP-Config (Seite 112)).

Synchronisationsart und Taktsynchronität einstellen für I-Device mit IRT

1. Doppelklicken Sie im Baugruppenträger auf den **Interface-Eintrag** um den Dialog **Eigenschaften Interface** aufzurufen.
2. Wählen Sie im Register **Synchronisation** als Synchronisationsart **Sync-Slave** und als RT-Klasse **IRT** aus.
3. Wählen Sie im Register **IO-Zyklus** unter Aktualisierungszeit den **Modus** automatisch und bei Taktsynchronität **IO-Device taktsynchron** zuordnen Servo.

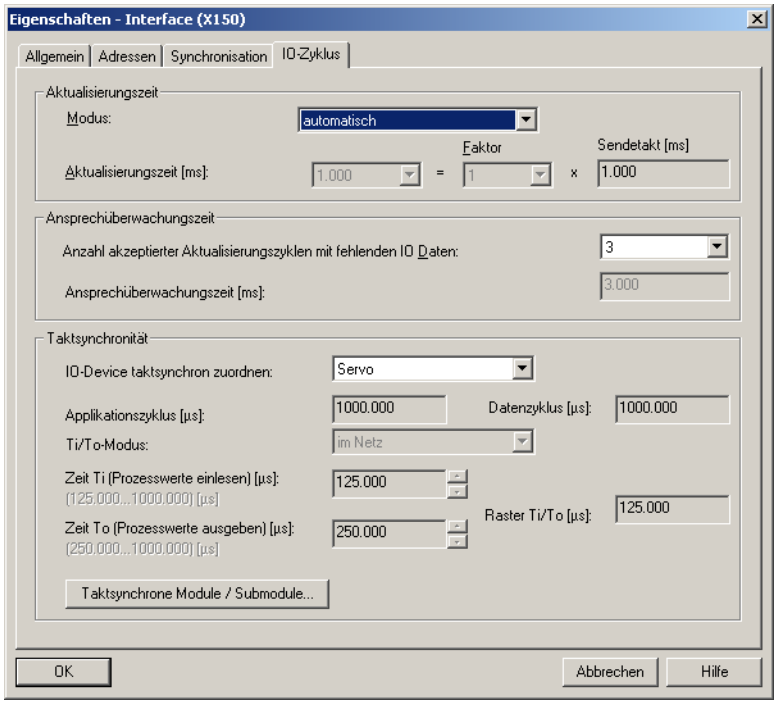


Bild 4-40 Taktsynchronität beim I-Device einstellen

Aktualisierungszeit und Sendetakt einstellen

I-Device mit RT

- Für I-Devices mit RT ist die Aktualisierungszeit einzustellen. Doppelklicken Sie dazu auf das PROFINET IO System und wählen Sie im Dialog **Eigenschaften PROFINET Subnetz** das Register **Aktualisierungszeit** aus. Stellen Sie dort die Aktualisierungszeit ein.

I-Device mit IRT

- Für I-Devices mit IRT ist der Sendetakt einzustellen. Der Sendetakt im Projekt des I-Devices muss gleich zum Sendetakt im Projekt des übergeordneten IO-Controllers eingestellt werden. Den Sendetakt des übergeordneten Projektes können Sie in HW Konfig über **Bearbeiten > PROFINET IO > Domain Management** einstellen.

4.5.7 I-Device-Stellvertreter löschen

Die GSD-Dateien für die I-Device-Stellvertreter unter Preconfigured Stations befinden sich in folgendem Verzeichnis:

<Program Files>\Siemens\Step7\S7DATA\GSD,
z. B. GSDML-V2.25-#Siemens-PreConf_**D455-2_IRT**-20100830-132044.xml.

D455-2_IRT ist dabei der Name des I-Device-Stellvertreters. Die I-Device-Stellvertreter können gelöscht werden, indem die betreffenden XML-Files gelöscht werden. Die Anzeige der I-Device-Stellvertreter unterhalb von Preconfigured Stations wird erst aktualisiert, wenn erneut eine GSD-Datei erstellt und installiert wird.

4.6 Kommunikationsprojektierung laden

4.6.1 PROFINET IO Projektierung laden

Voraussetzung

Es ist ein PG/PC angeschlossen, mit dem Sie ONLINE gehen können.

Vorgehensweise

Die Projektierungsdaten müssen nach erfolgter Projektierung von PROFINET IO in alle beteiligten Controller geladen werden.

1. Markieren Sie in NetPro das Ethernet-Subnetz und wählen den Menübefehl **Zielsystem > Laden im aktuellen Projekt > Stationen am Subnetz**.

4.7 Datenaustausch zwischen SIMATIC und SIMOTION über PROFINET

4.7.1 Datenaustausch durch Verwenden von I-Devices

Beschreibung

Die Kopplung zwischen SIMATIC und SIMOTION ist mit folgenden Funktionen möglich:

- TCP/ UDP*)-Anwenderkommunikation
- PROFINET IO/ RT, über S7-300 CP als Device

- PROFINET IO/ RT, über PN/PN Koppler
- PROFINET IO/ RT, über I-Device

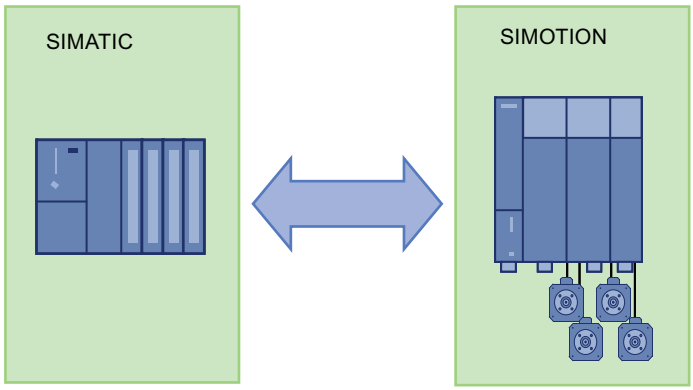


Bild 4-41 Datenaustausch zwischen SIMOTION und SIMATIC

Hinweis

In den SIMOTION Utilities & Applications ist ein FAQ zum Thema PROFINET RT I-Device-Kopplung zwischen einer SIMOTION-Steuerung und einer SIMATIC-Steuerung beschrieben. Die SIMOTION Utilities & Applications sind im Lieferumfang von SIMOTION SCOUT enthalten.

Die folgenden drei Anwendungsfälle werden betrachtet:

Fall A: Zwei getrennte Projekte, SIMATIC und die SIMOTION als I-Device in einem Projekt, SIMOTION als Controller im zweiten separaten Projekt.

Fall B: Ein Projekt für alle Komponenten.

Fall C: Mehrfachverwendung eines I-Devices.

Nähere Informationen zur Projektierung von I-Devices, siehe auch das Kapitel I-Device projektieren (Seite 118)

Beachten Sie auch den FAQ zur Projektierung FAQ I-Device (http://support.automation.siemens.com/WW/view/de/29578823)

4.7.2 PN-PN-Coupler

Beschreibung

Der PN/PN Coupler dient dazu, zwei PROFINET IO Systeme miteinander zu verbinden und Daten zwischen diesen auszutauschen. Die maximale Größe der übertragbaren Daten beträgt 256 Bytes Eingangsdaten und 256 Bytes Ausgangsdaten.

Der PN/PN Coupler hat als ein Gerät zwei PROFINET-Schnittstellen, die jeweils mit einem anderen Subnetz verbunden werden.

In der Projektierung werden aus einem PN/PN Coupler zwei IO-Devices abgeleitet, und zwar für jeden Controller mit ihrem Subnetz jeweils ein IO-Device. Der jeweils andere Teil des PN/PN Couplers wird als Koppel-Partner bezeichnet. Mit Abschluss der Projektierung werden die beiden Teile zusammengeführt.

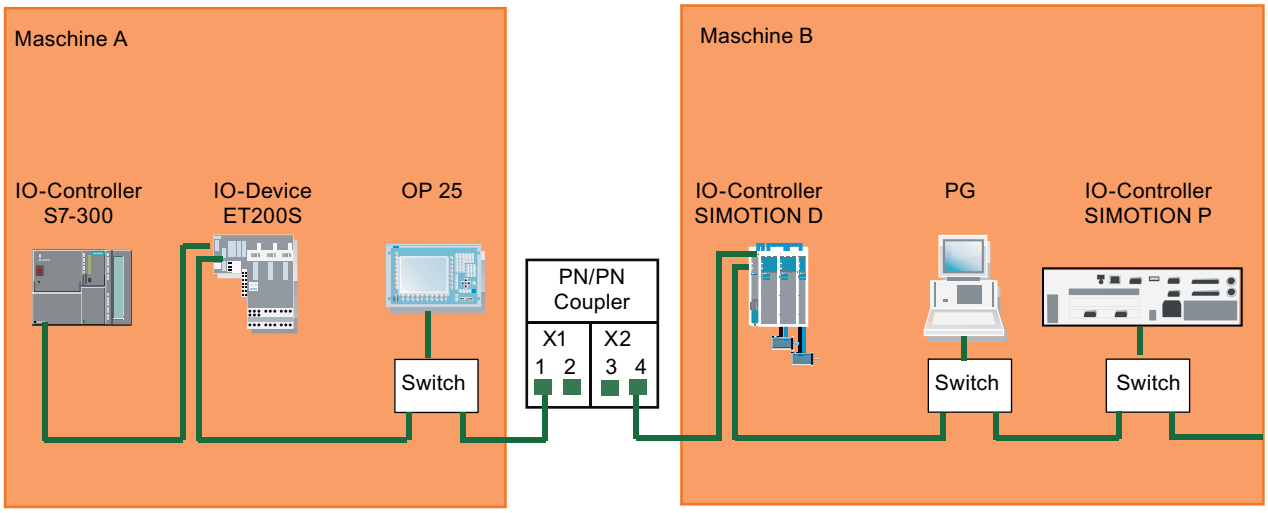


Bild 4-42 Kopplung zweier PROFINET-Subnetze mit einem PN/PN-Coupler

Die beiden Maschinen im Bild sind über den PN/PN-Coupler galvanisch getrennt. Soll es z. B. auch möglich sein, mit dem PG von Maschine B auf Maschine A online zu gehen, kann im PN/PN-Coupler eine Brücke zwischen Port 2 und 3 gesteckt werden. Damit geht die galvanische Trennung verloren.

Hinweis

Detaillierte Information über PN/PN-Coupler finden Sie in der entsprechenden Gerätedokumentation.

PN/PN-Coupler projektieren

Für die Projektierung des PN/PN-Couplers werden zwei PROFINET-Geräte angelegt, die linke (X1) und die rechte (X2) Seite. Den PN/PN-Coupler projektieren Sie mit HW Konfig. Wenn beide Subnetze in einem Projekt projektiert sind, können Sie mit STEP 7 den PN/PN-Coupler für beide Subnetze projektieren. Sind die Subnetze in verschiedenen Projekten projektiert, müssen Sie den Coupler in jedem Projekt projektieren.

Hinweis

Der PN/PN-Coupler wird zum Datenaustausch SIMOTION mit SIMATIC verwendet. Er ist aber auch für den schnellen Austausch von F-Signalen zwischen SIMATIC-CPU die bevorzugte Lösung.

4.7.3 Kommunikation über Standardprotokolle

Beschreibung

Mit TCP und UDP können Daten zwischen SIMOTION, SIMATIC, anderen Steuerungen und Fremdsystemen ausgetauscht werden. Die PROFINET-Schnittstelle ist eine Standard-Ethernet-Schnittstelle und unterstützt diese Protokolle. Das Anwenderprogramm muss dabei die Verwaltung der Kommunikationsverbindung realisieren. Über diese Verbindung können Sie z. B. Daten zwischen einem SIMATIC CPU und SIMOTION Controller über PROFINET austauschen.

UDP ist ein verbindungsloses Protokoll, d. h., es wird gesendet ohne Rückmeldung, ob der Empfänger die Nachricht erhalten hat. TCP ist ein verbindungsortientiertes Protokoll, d. h., es wird zuerst eine logische Verbindung aufgebaut. Erst wenn diese Verbindung steht, wird gesendet. Die Verbindung, d. h., der Empfang der Nachrichten, wird überwacht.

Datenaustausch über TCP:

- Verbindungsaufbau
- Datenverwaltung
- Verbindungsüberwachung
- Verbindungsabbau

Die Verwendung der Systembefehle ist ausführlich im Abschnitt **Ethernet Allgemein (TCP/IP- und UDP-Verbindungen)** beschrieben.

Siehe auch

Übersicht SIMOTION Systemfunktionen (Seite 158)

Übersicht SIMOTION Systemfunktionen (Seite 179)

4.8 Diagnose und Alarmverhalten

4.8.1 PROFINET IO Alarm- und Diagnosemeldungen an SIMOTION

Beschreibung

Für PROFINET IO ist eine Alarm- und Diagnosefunktionalität für die PROFINET-Geräte verfügbar.

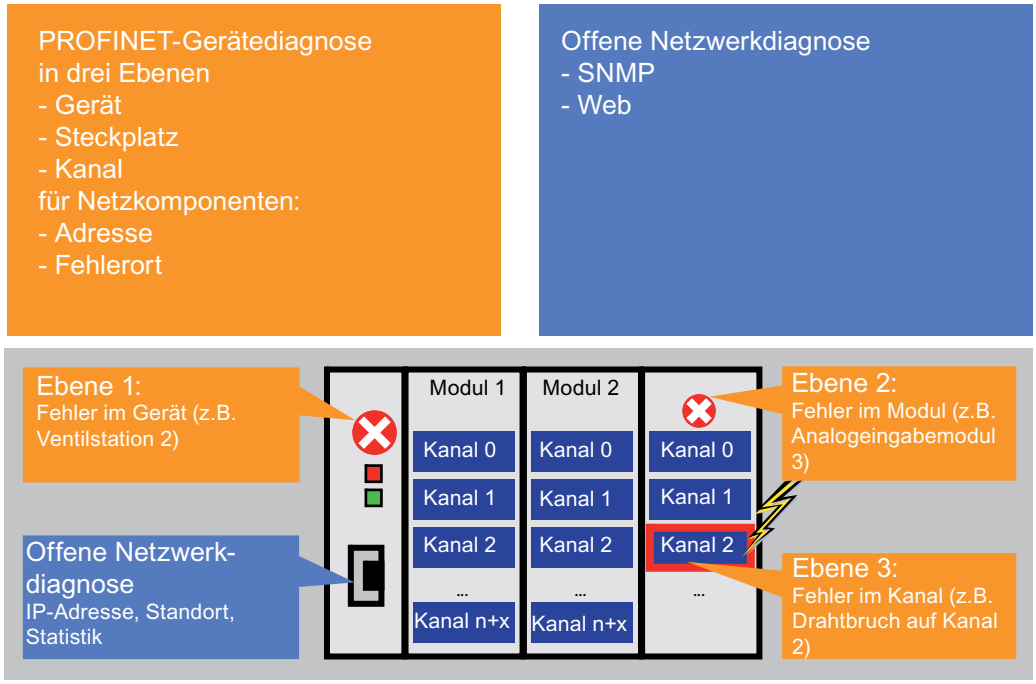


Bild 4-43 Diagnoseübersicht am Beispiel ET200

Gerätediagnose

Die Gerätediagnose kann in drei Ebenen aufgeteilt werden. Detaillierte Informationen finden Sie im Abschnitt Diagnosemodell.

4.8.2 Diagnosemodell

PROFINET IO setzt auf vollständig standardisierte Diagnosemechanismen. Dies ist für eine herstellerübergreifende Geräte- und System-Diagnose äußerst hilfreich.

Aufgrund der größeren Mengengerüste ist es nicht möglich, die Statusinformationen aller Stationen im IO-Controller zu halten. Deshalb werden nur aktuelle Diagnose-Ereignisse über standardisierte Alarme an den IO-Controller übertragen.

Durch die Nutzung eines quittierten Dienstes wird die Übertragung der Diagnoseereignisse in kausaler Reihenfolge ermöglicht. Der Status einer Station wird von dieser gespeichert und kann von einem Diagnosesystem jederzeit und direkt über standardisierte Datensätze ausgelesen werden, siehe entsprechende Dokumentation zu STEP7 und Dokumentation zur Systemfunktion _readRecord( ).

Zugriff auf Alarm- und Diagnosedaten

Für PROFINET IO wird zwischen folgenden Alarm- bzw. Diagnosemeldungen unterschieden:

- Alarme, die von IO-Devices an den IO-Controller geschickt werden
- Alarme, die im IO-Controller auftreten

Die folgende Grafik zeigt die Zugriffsmöglichkeiten auf Diagnosedaten:

1. Diagnose auf PG
   Das PG liest die Diagnose direkt vom IO-Device. Die Visualisierung findet im PG statt.
2. Diagnose in der Steuerung
   Das IO-Device sendet die Diagnose an den IO-Controller, die Reaktion auf die Störung findet in der Steuerung statt.

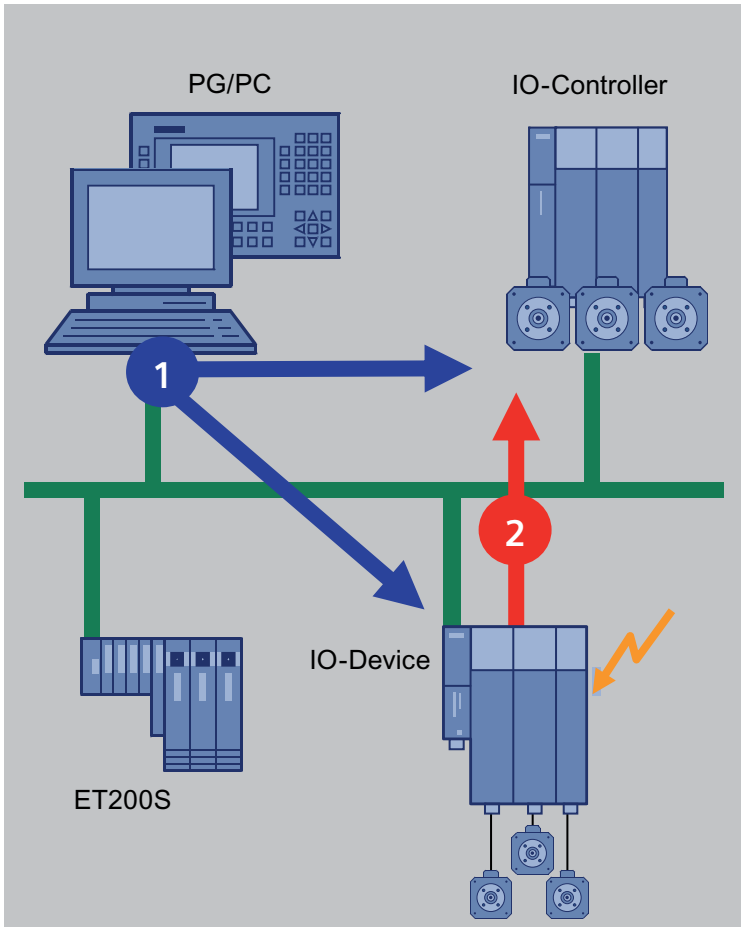


Bild 4-44 Zugriff auf Diagnosedaten

4.8.3 Alarme am IO-Controller

Beschreibung

Im IO-Controller werden eine Reihe von Alarmen ausgegeben. Die auftretenden Alarme werden mit der betreffenden EventID im Diagnosepuffer von SIMOTION aufgelistet. Folgende Alarme können unterschieden werden:

- Alarme beim direkten Datenaustausch zwischen IO-Controllern
- Stationsalarme, die von der PROFINET-Schnittstelle gemeldet werden

Folgende Tabelle zeigt die Alarme von PROFINET IO, wie sie in SIMOTION abgebildet werden:

Alarm (TSI#InterruptId)	TSI#eventClass	TSI#faultId	Bedeutung
Stationsausfall (_SC_STATION_DISCONNECTED (= 202))	16#39	16#CA	Systemfehler PROFINET IO; Hierfür gibt es nur ein kommendes Ereignis, ein gehendes Ereignis wird auf 16#38 - 16#CB abgebildet und zwar für jedes vorhandene IO-Device.
		16#CB	Stationsausfall eines IO-Device
		16#CC	IO-Device gestört. Kanaldiagnose oder herstellerspezifische Diagnose steht an.
Stationswiederkehr (_SC_STATION_RECONNECTED (= 203))	16#38	16#CB	Stationswiederkehr eines IO-Device ohne Fehler
		16#CC	Störung der IO Device behoben
		16#CD	Stationswiederkehr eines IO-Device, aber Fehler: Sollaufbau <> Istaubau
		16#CE	Stationswiederkehr eines IO-Device, aber Fehler bei Baugruppenparametrierung

Verwendung der TaksStartInfo

Informationen zum Verwenden der TaskStartInfo für die PeripheralFaultTask finden Sie im Handbuch **Basisfunktionen**.

4.8.4 Alarme vom IO-Device an den IO-Controller

Beschreibung

Die Alarme werden über den PROFINET-Alarmmechanismus vom IO-Device zu seinem zugehörigen IO-Controller übertragen. Die Alarme werden in den Diagnosepuffer eingetragen und können über die PeripheralFaultTask ausgewertet werden. Folgende Tabelle zeigt, wie die Alarme auf die PeripheralFaultTask abgebildet werden.

Alarm (TSI#InterruptId)	TSI#event Class	TSI#faultId	Bedeutung
Diagnosis (kommend)	16#39	16#42	Kommender Diagnosealarm.
Diagnosis disappears (gehend) Multicast Communication Mismatch Port Data Change Notification Sync Data Changed Notification Isochronous Mode Problem Notification Network component problem notification (_SC_DIAGNOSTIC_INTERRUPT (=201))	16#38	16#42	Gehender Diagnosealarm
Prozessalarm (_SC_PROCESS_INTERRUPT (= 200))	16#11	16#41	Prozessalarm
Pull Alarm	16#39	16#51	PROFINET IO Modul wurde entfernt oder kann nicht adressiert werden.
		16#54	PROFINET IO Submodul wurde entfernt oder kann nicht adressiert werden.
Plug Alarm Plug Wrong Submodule Alarm Return of Submodule Alarm (_SC_PULL_PLUG_INTERRUPT (=216))	16#38	16#54	PROFINET IO Modul oder Submodul wurde gesteckt, Modultyp OK (Istaufbau = Sollaufbau)
		16#55	PROFINET IO Modul oder Submodul wurde gesteckt, aber falscher Modultyp (Istaufbau <> Sollaufbau)
		16#56	PROFINET IO Modul oder Submodul wurde gesteckt, aber Fehler bei der Baugruppenparameterierung
		16#58	IO Status eines Moduls hat sich von BAD nach GOOD geändert
Status			Nicht unterstützt
Update			Nicht unterstützt
Time data changed notification			Nicht unterstützt
Upload and storage notification			Nicht unterstützt
Pull module			Nicht unterstützt
Manufacturer specific			Nicht unterstützt
Profile specific			Nicht unterstützt

Alarmtypen, die mit "Nicht unterstützt" gekennzeichnet sind, werden vom SIMOTION-Controller mit "not supported" quittiert und nicht in den Diagnosepuffer eingetragen.

Verwendung der TaksStartInfo

Informationen zum Verwenden der TaskStartInfo für die PeripheralFaultTask finden Sie im *Handbuch SIMOTION Basisfunktionen*.

Diagnosedaten übertragen

Der genaue Grund für den Alarm wird in Form von Diagnosedaten bereitgestellt. Diese können über die Funktion _readDiagnosticData() ausgelesen werden. Die Länge ist auf 240 Byte beschränkt.

Ab V4.2 können Sie mit dem Funktionsbaustein _readVariableDiagnosticData() durch das Anwenderprogramm Diagnosedaten einer Station/Moduls bis zu 65535 Bytes auslesen.

Programmbeispiel für das Auslesen von TSI#-Infos der PeripheralFaultTask

```
; ....
; Variablendeklaration
    PHERIPHERIE_Alarminfo   : STRUCT
                                ALH_internalID      : UDINT;
                                ALH_EingangsAdresse : DINT;
                                ALH_AusgangsAdresse : DINT;
                                ALH_DiagnoseAdresse : DINT;
                                ALH_Details         : DWORD;
                                ALH_Starttime       : DATE_AND_TIME;
                                ALH_EventClass      : UINT;
                                ALH_FaultID         : UINT;
; ....
; Auslesen der Peripherie Fault Task Infos:
  P_ALH_Info[p_alhcount].ALH_internalID      := TSI#interruptID;
  P_ALH_Info[p_alhcount].ALH_EingangsAdresse := TSI#logbaseadrin ;
  P_ALH_Info[p_alhcount].ALH_AusgangsAdresse := TSI#logbaseadrout ;
  P_ALH_Info[p_alhcount].ALH_DiagnoseAdresse := TSI#logdiagadr ;
  P_ALH_Info[p_alhcount].ALH_Details         := TSI#details ;
  P_ALH_Info[p_alhcount].ALH_Starttime       := TSI#starttime ;
  P_ALH_Info[p_alhcount].ALH_EventClass      := TSI#eventClass ;
  P_ALH_Info[p_alhcount].ALH_FaultID         := TSI#faultID ;
; ....
```

4.8.5 Alarme bei direktem Datenaustausch zwischen IO-Controllern

Beschreibung

Bei PROFINET IO mit IRT gibt es eine Kommunikationsüberwachung zwischen IO-Controllern. Wenn diese feststellt, dass keine IRT-Daten mehr empfangen werden - entweder es kommen überhaupt keine Daten oder kommen zu spät - wird ein Stationsausfallalarm generiert. Wenn eine Wiederaufnahme der Kommunikation festgestellt wird, so wird ein Stationswiederkehralarm generiert. Kommen IRT-Daten dreimal zu spät, wird ein Stationsausfallalarm gemeldet.

Hinweis

Bei der Kommunikationsüberwachung wird nur der Empfänger-Slot überwacht.

Folgende Tabelle zeigt die Alarme von PROFINET IO zwischen IO-Controllern des direkten Datenaustauschs, wie sie in SIMOTION abgebildet werden:

Alarm (TSI#InterruptId)	TSI#eventClass	TSI#faultId	Bedeutung
Stationsausfall (_SC_STATION_DISCONNECTED (= 202))	16#39	16#F3	Der Empfänger des direkten Datenaustauschs empfängt keine Daten mehr.
Stationswiederkehr (_SC_STATION_RECONNECTED (= 203))	16#38	16#F0	Der Sender des direkten Datenaustauschs ist hochgelaufen und sendefähig.
		16#F1	Der Empfänger des direkten Datenaustauschs ist ohne Fehler hochgelaufen und Empfänger empfängt wieder Daten (alle Empfangsbereiche verfügbar).
		16#F2	Der Empfänger des direkten Datenaustauschs ist mit Fehler hochgelaufen und der Empfänger empfängt wieder Daten (mind. ein Empfangsbereich nicht verfügbar)

4.8.6 Alarme von SINAMICS S120 Antrieben

Beschreibung

Alarme, die vom SINAMICS S120 CU320/CBE20 bzw. SINAMICS S120 CU310 PN ausgelöst werden, werden über den PROFINET-Alarmkanal abgesetzt. Bei den Alarmen sind zwei Typen zu unterscheiden:

- Alarme, die vom PROFINET-Interface ausgelöst werden und im unmittelbaren Zusammenhang mit PROFINET stehen.
- Alarme, die von der Applikation/Technologie im Antrieb ausgelöst werden.

PROFINET-Alarme

Folgende Alarme werden über die PROFINET-fähige SINAMICS Baugruppe unterstützt:

Alarm	Beschreibung
Port Data Change Notification	Detaillierte Beschreibung finden Sie unter Alarme am IO-Controller (Seite 139)
Sync Data Changed Notification	
Isochronous Mode Problem Notification	
Multicast Communication Mismatch	

Alarme der Technologie/Applikation

Alarme werden nicht als Standard-PROFINET-Alarme an den Controller geschickt, sondern als PROFIdrive Alarm abgebildet. Der Alarmpuffer kann über einen Parameterauftrag abgeholt werden (Trigger im Statuswort/Zustandswort Bit7).

4.8.7 Systemfunktionen für die Diagnose für PROFINET bzw. PROFIBUS

Übersicht über System- und Diagnosefunktionen

Die folgende Tabelle zeigt eine Übersicht über die verschiedenen System- und Diagnosefunktionen für PROFINET IO. Unterschiede zu PROFIBUS DP werden ebenfalls aufgezeigt.

Detaillierte Informationen zu den jeweiligen Funktionen finden Sie in der Referenzliste der Funktionen *Listenhandbuch, SIMOTION Systemfunktionen/-variablen Geräte*.

Funktion	Anmerkung	PROFIBUS	PROFINET
_getStateOfSingleDpSlave()	Die Funktion ermittelt den Zustand der Kommunikation zu einem zyklischen Kommunikationspartner (Device - PROFINET, Slave - PROFIBUS, Controller-Controller Querverkehr Sender oder Empfänger - PROFINET).	Logische Diagnoseadresse des DP-Slaves	Logische Diagnoseadresse des IO Devices
_getStateOfDpSlave()	_getStateOfDpSlave liefert die Information, ob der PROFIBUS DP-Slave bzw. das PROFINET IO Device aktiviert oder deaktiviert ist.	Logische Diagnoseadresse DP-Slave	Logische Diagnoseadresse des IO Devices

Funktion	Anmerkung	PROFIBUS	PROFINET
_getStateOfAllDPStations()	Die Systemfunktion ermittelt den Zustand der Kommunikation zu zyklischen Kommunikationspartnern (Device - PROFINET, Slave - PROFIBUS, Controller-Controller Querverkehr Sender oder Empfänger - PROFINET). Neben dem Aktivierungsstatus erhalten Sie auch die Information, ob er überhaupt verfügbar ist.	Logische Diagnoseadresse der Schnittstellen	Logische Diagnoseadresse der Schnittstellen Ab 4.2 Diese Systemfunktion bildet bei PROFINET-IO-Devices ein Summensignal aus den in dem Device vorhandenen Modulen – damit signalisiert die Funktion dem Anwenderprogramm auch gezogene bzw. gestörte Module in dem Device-Rückgabewerte.
_getStateOfIO()	Diese Funktion stellt den Status der DP-Stationen, der Module und Submodule dem Anwenderprogramm bereit. Weiterhin liefert sie diese Informationen auch detailliert zu Modulen bzw. Submodulen. Unterstützt werden: • Interfaces wie PROFIBUS-DP oder PROFINET • Stationen (Slaves, Devices), Module oder Submodule Diese Funktionalität liefert eine logische Diagnose-Adresse bzw. logische I/O-Adresse identifizierten Modulen eine Liste der zugeordneten Submodule.		
_getNextLogAddress()	Über diese Funktion können alle projektierten logischen Adressen eines Segments ermittelt werden.	Logische Diagnoseadresse	Logische Diagnoseadresse

Funktion	Anmerkung	PROFIBUS	PROFINET
_readDiagnosticData() _getStateOfDiagnosticDataCommand()	Realisiert das Auslesen der Diagnosedaten eines DP-Slaves durch das Anwenderprogramm. Bei PROFIBUS wird die Diagnose für den Slave ausgelesen, d. h. die komplette Diagnose-Information vom Slave wird geliefert. Der Aufbau der Diagnosedaten ist in IEC 61158-6-3 beschrieben. Bei PROFINET wird die Subslot spezifische Diagnose gelesen (d. h. der Datensatz 0x800A). Es wird die Diagnose von einem Subslot geliefert. Der Aufbau der Diagnosedaten ist in IEC 61158-6-10 beschrieben.	Logische Diagnoseadresse DP-Slave	Logische Diagnoseadresse des Devices, des Modul/Slot oder IO-Adresse des Kanals/Subslot
_readVariableDiagnosticData()	Der Funktionsblock realisiert das Auslesen der Diagnosedaten einer Station bzw. eines Moduls durch das Anwenderprogramm. Das Diagnoseformat für PROFINET IO V2.2 und PROFIBUS DP ist in der Norm IEC 61158-6 beschrieben.	-	Mit dem FB _readVariableRecord können Datensätze mit einer Gesamtlänge > 240 Bytes gelesen werden. Diese Funktion kann für PROFIBUS und PROFINET genutzt werden. Nur bei PROFINET werden vom Device Datensätze > 240 Bytes zurückgeliefert.
_readDriveFaults() nur SINAMICS (ET200FC, S120, etc.)	Ermöglicht das Lesen des aktuellen Störpuffereintrages im Antrieb.	Logische Basisadresse des Antriebes (Slot)	Jede gültige logische I/O-Adresse des betreffenden Subslots oder Diagnose-adresse des PAP (für Module ohne zyklische Daten)

4.8.8 PROFINET Gerätediagnose in STEP 7

Gerätediagnose in STEP 7

Über HW Konfig kann im SCOUT eine ONLINE-Gerätediagnose über PROFINET durchgeführt werden. Die Diagnose liefert neben Steckplatz, Kanalnummer auch die Fehlerart. Die Diagnose funktioniert analog zu PROFIBUS.

Vorgehensweise

1. Gehen Sie ONLINE und rufen Sie HW Konfig für das entsprechende SIMOTION Gerät auf.
2. Wählen Sie **Zielsystem > Ethernet Teilnehmer diagnostizieren, beobachten/steuern** aus. HW Konfig sucht nach allen Netzwerkteilnehmern. Das Fenster **(Diagnose) ONLINE** wird eingeblendet und zeigt die Netzwerkteilnehmer an.
3. Klicken Sie mit der rechten Maustaste auf den gewünschten Teilnehmer und wählen Sie **Eigenschaften** aus. Die Detaildiagnose wird eingeblendet. Hier wird der entsprechende Fehler angezeigt.

4.8.9 Diagnoselarme PROFINET IO und DS0

4.8.9.1 Wartungskonzept Diagnosealarm PROFINET IO

Beschreibung

Ein Gerät bzw. eine Baugruppe eines Automatisierungssystemes kann grundsätzlich die Zustände Gut (Good) oder Störung (Failure) haben. Um die Verfügbarkeit von Sensoren/Aktoren, Geräten und Baugruppen zu erhöhen, werden von diesen Komponenten selbst, neben diesen beiden Zuständen, zusätzlich Informationen zu einer erforderlichen Wartung geliefert. Diese zusätzlichen Informationen umfassen Angaben zum Wartungszustand, Verschleißzustand, Restlebensdauer etc. Man spricht hier von einem erweiterten Wartungskonzept. Ziel des erweiterten Wartungskonzepts ist das frühzeitige Erkennen und Beseitigen von potenziellen Störungen - noch bevor es zum Produktionsausfall kommt. Dazu werden die beiden Zustände Gut bzw. Störung um die Zustände Wartungsanforderung (Maintenance required) bzw. Wartungsbedarf (Maintenance demanded) erweitert.

Die genannten Zustände lassen sich auf das folgende Wartungszustandsmodell abbilden.

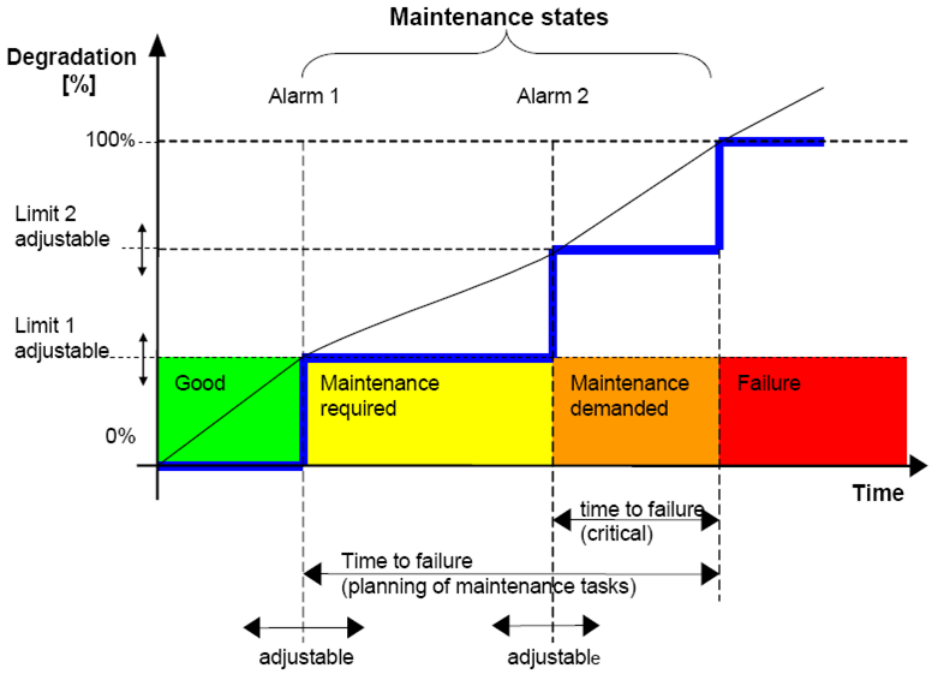


Bild 4-45 Wartungszustandsmodell PROFINET IO

Wartungszustandsmodell

- Good (grün): Gut
- Maintenance required (gelb): Wartungsanforderung (Wartung ist einzuplanen)
- Maintenance demanded (orange): Wartungsbedarf (Wartung ist durchzuführen)
- Failure (rot): Störung

4.8.9.2 Gerätemodell IO-Device

Beschreibung

Im Gerätemodell von PROFINET IO ist für ein IO-Device definiert, dass sich dieses in Slots und Subslots unterteilt. Auf einen Slot wird ein Modul (= physikalische Baugruppe) und auf einen Subslot ein Submodul (= physikalische Sub-Baugruppe) abgebildet.

Jedes Submodul kann Träger folgender Objekte sein:

- Zyklische Daten (Nutzdaten der E/A-Schnittstelle zur Steuerung des Prozesses)
- Alarme (z.B. Diagnosealarm)
- Datensätze

- Parameter
- Diagnose

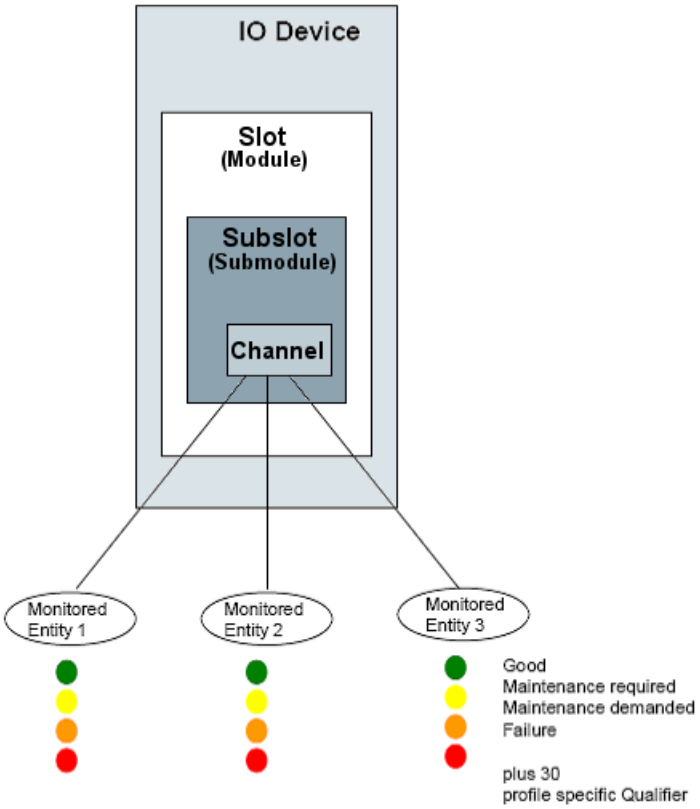


Bild 4-46 Einordnung des Kanals, der Diagnosezustände und der Sammelinfos

Unterhalb von Submodulen sind im Gerätemodell von PROFINET IO für die Diagnose Kanäle (Channel) definiert. Ein Kanal ist eine logische Unterstruktur eines Submodules. Ein Submodul kann bis zu 65536 Kanäle besitzen. Innerhalb der Kanäle werden definierte Diagnosefunktionen (z.B. Kurzschluß, Drahtbruch, Übertemperatur) überwacht. Ein Kanal kann auch mehrere Diagnosefunktionen parallel überwachen.

Als Ergebnis aus diesen Überwachungen entsteht eine Kanal-Diagnose. Mehrere unterschiedliche Kanal-Diagnosen in einem Submodul können gleichzeitig anstehen. Es können natürlich auch Kanal-Diagnosen von verschiedenen Submodulen anstehen. Kanal-Diagnosen werden über einen Diagnosealarm gemeldet. Die Meldung des Diagnosealarms erfolgt für jedes Submodul, in dem eine Kanal-Diagnose ansteht.

4.8.9.3 Diagnoselarme PROFINET IO und DS0

Beschreibung

Beim Auftreten eines diagnoserelevanten Ereignisses (z.B. Störung oder Wartungsanforderung) in einem IO-Device generiert dieses einen Diagnosealarm und sendet diesen an den zugehörigen IO-Controller. Das als IO-Controller fungierende SIMOTION Gerät meldet dann einen Diagnosealarm über die PeripheralFaultTask mit TSI#interruptId = _SC_DIAGNOSTIC_INTERRUPT (= 201). Unter der TSI#InterruptId ist der Datensatz 0 (DS0) für den Diagnoselarm zu finden. Der Datensatz 0 (DS0) liefert die Sammelzustände der Kanäle (Channel) eines Submodules. Ein Diagnoselarm kann mit TSI#eventClass = 0x38, TSI#faultId = 0x42 als kommender Diagnosealarm und mit TSI#eventClass = 0x39, TSI#faultId = 0x42 als gehender Diagnosealarm auftreten.

Ein Diagnosealarm besitzt für die Summe aller Diagnosefunktionen der Kanäle (Channel) innerhalb eines Submodules im Datensatz 0 (DS0) folgende Sammelzustände in Form von folgenden Sammelbits:

- Sammelstörung: DS0.Byte0.Bit 0
- Sammelwartungsanforderung: DS0.Byte1.Bit 7
- Sammelwartungsbedarf: DS0.Byte2.Bit7

Die Sammelbits werden aus der logischen Veroderung der jeweiligen Einzelbits aller Diagnosefunktionen der Kanäle eines Submodules gebildet. Die drei Sammelbits sind unabhängig voneinander und beeinflussen sich nicht gegenseitig. Die Sammelbits werden damit unabhängig voneinander gesetzt bzw. rückgesetzt.

Tritt nur eine Änderung der Wartungszustände auf, einzelne Wartungsanforderungen bzw. Wartungsbedarfe stehen noch an im Submodul, und keine Störung, dann wird ein kommender Diagnoselarm gemeldet, und die Sammelbits Sammelwartungsanforderung/Sammelwartungsbedarf im DS0 entsprechend gesetzt. Sind alle Wartungszustände verschwunden und es steht keine Störung an, dann wird ein gehender Diagnosealarm gemeldet, und die beiden Sammelbits Sammelwartungsanforderung/Sammelwartungsbedarf auf 0 gesetzt. Generell gilt somit die Regel, sobald mindestens eines der Sammelbits Sammelstörung oder Sammelwartungsanforderung oder Sammelwartungsbedarf auf 1 gesetzt ist, wird ein kommender Diagnosealarm gemeldet. Sobald jedoch keines der Sammelbits Sammelstörung oder Sammelwartungsanforderung oder Sammelwartungsbedarf mehr auf 1 gesetzt ist, dann wird ein gehender Diagnosealarm gemeldet.

Erklärung der Bits des Datensatzes DS0

Tabelle 4- 6 Tabelle: DS0 Byte 0

Bit	Beschreibung	Kommentar
Bit 0	Module Fault/OK	Sammelstörung (Diagnose) für ein Submodul: • 0: Sammelstörung (Diagnose) liegt nicht vor • 1: Sammelstörung (Diagnose) liegt vor
Bit 1	Internal error	Immer 0
Bit 2	External error existent	Ist inhaltlich gleich zu Bit 0

Bit	Beschreibung	Kommentar
Bit 3	Channel error existent	Ist inhaltlich gleich zu Bit 0
Bit 4	External auxiliary power missing	Immer 0
Bit 5	Front connector missing	Immer 0
Bit 6	Module not parameterized	Immer 0
Bit 7	Wrong parameters in module	Immer 0

Tabelle 4- 7 Tabelle: DS0 Byte 1

Bit	Beschreibung	Kommentar
Bit 0-3	Type class of module	3: Die Typ-Klasse 3 ist als dezentrale Peripherie zu verstehen und umfasst damit auch PROFINET IO.
Bit 4	Channel information existent	0: keine auslesbaren Kanalinformationen vorhanden 1: auslesbare Kanalinformationen sind vorhanden in den Alarmdaten des Diagnosealarmes. Die Alarmdaten können über die Systemfunktion _readDiagnosticData ausgelesen werden.
Bit 5	User information existent	0: keine Kanaldiagnose bzw. herstellerspezifische Diagnose vorhanden 1: mindestens eine Kanaldiagnose und/oder herstellerspezifische Diagnose ist vorhanden
Bit 6	Diagnosis alarm from proxy	Always 0
Bit 7	Maintenance Required	Sammelwartungsanforderung für ein Submodul: • 0: Sammelwartungsanforderung liegt nicht vor • 1: Sammelwartungsanforderung liegt vor

Tabelle 4- 8 Tabelle: DS0 Byte 2

Bit	Beschreibung	Kommentar
Bit 0	-	Immer 0
Bit 1	-	Immer 0
Bit 2	-	Immer 0
Bit 3	-	Immer 0
Bit 4	-	Immer 0
Bit 5	-	Immer 0
Bit 6	-	Immer 0
Bit 7	Maintenance Demanded	Sammelwartungsbedarf für ein Submodul: • 0: Sammelwartungsbedarf liegt nicht vor • 1: Sammelwartungsbedarf liegt vor

Tabelle 4- 9 Tabelle: DS0 Byte 3

Bit	Beschreibung	Kommentar
Bit 0	-	Immer 0
Bit 1	-	Immer 0
Bit 2	-	Immer 0
Bit 3	-	Immer 0
Bit 4	-	Immer 0
Bit 5	-	Immer 0
Bit 6	-	Immer 0
Bit 7	-	Immer 0

Werden detaillierte Informationen zu den Kanälen eines Submodules im Diagnosealarm benötigt, so müssen mit der Systemfunktion _readDiagnosticData die Alarmdatendaten zum Diagnosealarm ausgelesen und entsprechend ausgewertet werden.

Ethernet Allgemein (TCP- und UDP-Verbindungen) 5

5.1 Ethernet-Schnittstellen

5.1.1 Übersicht Ethernet

Übersicht

Nachfolgend wird beschrieben, wie TCP- und UDP-Ethernet-Verbindungen zwischen Kommunikationspartner projektiert werden können. TCP und UDP basieren auf Ethernet und dem IP-Protokoll.

5.1.2 Eigenschaften der SIMOTION Ethernet-Schnittstellen

Ethernet-Schnittstellen

SIMOTION besitzt je nach Gerät eine oder zwei Onboard-Ethernet-Schnittstellen. An die 8-poligen RJ45-Buchsen können Sie ein Industrial Ethernet mit einer Übertragungsgeschwindigkeit von 10/100 MBit/s bzw. 1000 MBit/s bei D4x5-2 DP/PN anschließen.

Bei Baugruppen mit 2 Ethernet-Schnittstellen gibt es keine HUB/Switch-Funktionalität, d. h., Telegramme werden nicht von einer Schnittstelle zur anderen weitergeleitet. Die Schnittstellen gehören zu getrennten Ethernet-Subnetzen. Die SIMOTION Geräte haben keine IP-Routerfunktionalität, sie leiten die Telegramme nicht von einem Subnetz zum anderen.

TCP/IP-Timeout-Parameter sind bei 2 Schnittstellen einmal für beide Schnittstellen einstellbar. Die Übertragungsrate/Duplex ist bei 2 Schnittstellen für beide Schnittstellen getrennt einstellbar. Es werden "Dienste über TCP" unterstützt für beide Ethernet-Schnittstellen, darüber ist ein S7-Routing von den Ethernet-Schnittstellen in die Profibus-Schnittstellen hinein möglich. Die "Dienste über TCP/IP" werden nicht von der einen Ethernet-Schnittstelle in die andere geroutet.

Alternativ können Sie ein Industrial Ethernet auch über die PROFINET-Baugruppen, wie z. B. CBE30 einer SIMOTION D4x5, anschließen (100 MBit/s).

Die MAC-Adressen der Ethernet-Schnittstellen sind außen auf dem Gehäuse sichtbar.

5.1.3 Verwendung der Ethernet-Schnittstelle

Verwendung der Ethernet-Schnittstelle

- zur Kommunikation mit STEP7, SIMOTION SCOUT und SIMATIC NET OPC über ein PG/PC
- zur Kommunikation über UDP (User Datagram Protocol) mit anderen Komponenten, z. B. anderen SIMOTION Geräten, SIMATIC Geräten oder PCs
- zur Kommunikation über TCP (Transfer Control Protocol) mit anderen Komponenten, z. B. anderen SIMOTION Geräten, SIMATIC S7 Stationen oder PC
- zum Anschließen von SIMATIC HMI Geräten, wie z. B. MP277, MP370 oder PC-basierte HMIs
- zur Kommunikation mittels SIMOTION IT DIAG und SIMOTION IT OPC XML-DA (ab V4.2 keine Lizenz notwendig)
- zur Kommunikation mittels SIMOTION VM (eigene Lizenz erforderlich)

5.2 Kommunikations-Bibliothek LCom

LCom Bibliothek

Die Bibliothek **LCom** basiert auf TCP, und vereinfacht die Anwendung der SIMOTION Systemfunktionen sowie der SIMATIC Kommunikationsbausteine, um eine Kommunikation zwischen mehreren Maschinen zu realisieren.

Hinweis

Die Bibliothek LCom finden Sie auf der SIMOTION SCOUT DVD "Documentations, Utilities&Applications". Auf der DVD sind ein Beispielprojekt und eine ausführliche Dokumentation zur Bibliothek enthalten.

Steuerungstypen

Folgende Steuerungstypen und Kombinationen werden von der Bibliothek LCom unterstützt:

- SIMOTION ↔ SIMOTION
- SIMATIC ↔ SIMATIC
- SIMOTION ↔ SIMATIC

Funktionen

- Die Sende- und Empfangsdaten müssen vom Datetyp BYTE sein. Außerhalb des *FBLComMachineCom* können beliebige Anwender-Strukturen via Marshalling in ein ARRAY OF BYTES umgewandelt werden.
- Bidirektionaler Betrieb
  - Es wird eine logische Punkt-zu-Punkt-Verbindung zwischen zwei Steuerungen aufgebaut.
  - Jede Steuerung kann über eine Verbindung gleichzeitig senden und empfangen.
- Konfigurationsabgleich der Kommunikationspartner (z. B. Sendetakt)
  - Zuweisung der Kommunikationsparameter an den Kommunikationspartner
  - Änderung der Konfiguration im laufenden Betrieb
- Arten der Datenübertragung
  - Zyklische Übertragung (Übertragung in einem festen zeitlichen Abstand)
  - Senden bei Datenänderung
  - Einmaliges Senden
- Senden und Empfangen von max. 64 kB Nutzdaten
- Quittierung der empfangenen Daten und deren Überwachung
- Die Anzahl der Telegramme, die unquittiert übertragen werden können, kann variieren (einstellbares *u8SlidingWindow*). Der LCom Baustein für die SIMATIC Steuerung unterstützt nur *u8SlidingWindow* = 1.
- Lebenszeichen-Überwachung
- Uhrzeitsynchronisation

5.3 TCP-Kommunikation

5.3.1 Übersicht TCP-Kommunikation

Kommunikation mit TCP (Transfer Control Protocol)

Die Kommunikation über TCP ist verbindungsorientiert, d. h., erst wenn eine Verbindung zum Kommunikationspartner aufgebaut ist, können Daten übertragen werden.

Eine Kommunikations-Verbindung ist charakterisiert durch zwei Endpunkte. Ein Endpunkt ist ein geordnetes Paar aus IP-Adresse und Port. Der Port am Client kann gleich/ungleich dem Port am Server sein. Dabei stellt in allgemeinen ein Endpunkt einen Server und der andere Endpunkt einen Client dar. Beim TCP Verbindungsaufbau muss es mindestens einen Client und einen Server geben.

Die Client-Server-Beziehung gilt nur bis zum Abschluss des Verbindungsaufbaus. Nach dem Verbindungsaufbau sind beide Kommunikationspartner gleichwertig, d. h. jeder der beiden kann zu einem beliebigen Zeitpunkt senden oder empfangen oder die Verbindung schließen.

Prinzip der TCP-Kommunikation (siehe Bild)

- Server setzt sich an Port wartend (1)
- Client meldet Verbindungsanforderung auf diesen Port an (2). Ist serverseitig kein Port angemeldet, wird mit TimeOut (Systemeinstellung) gewartet
- Server legt bei Verbindungsanmeldung internen Kommunikationsport an und gibt den Serverport für neuen Verbindungsaufbau frei. Der interne Kommunikationsport wird über die connectionId identifiziert (3)
- Daten senden / empfangen, sowohl vom Client wie auch vom Server über diese Verbindung möglich (4)
- am Serverport können weitere Verbindungen aufgebaut werden (5)
- eine bestehende Verbindung kann client- oder serverseitig geschlossen werden (6)
- Serverport für Verbindungsaufbau wird serverseitig geschlossen (7)

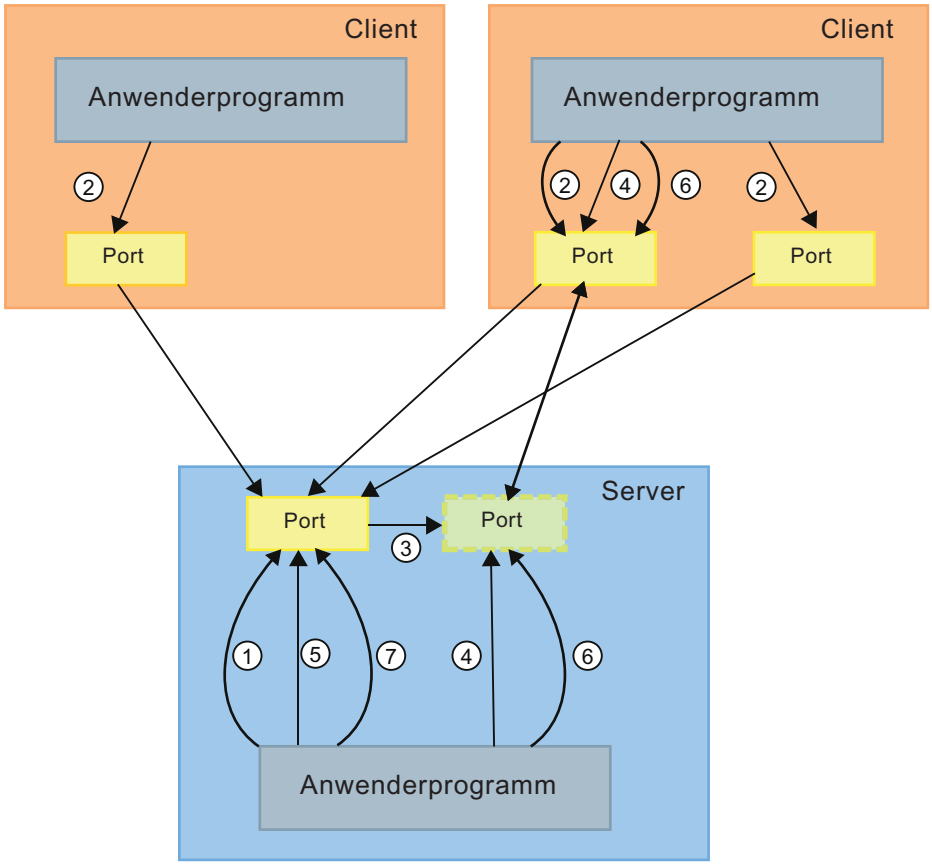


Bild 5-1 Prinzipieller Ablauf der Kommunikations-Sequenz einer TCP-Kommunikation

Prinzipielle Portvergabe:

- Der Port am Client kann gleich dem Port am Server sein.
- Der Port am Client kann ungleich dem Port am Server sein.

Datenaustausch zwischen Anwenderprogramm und TCP-Stack

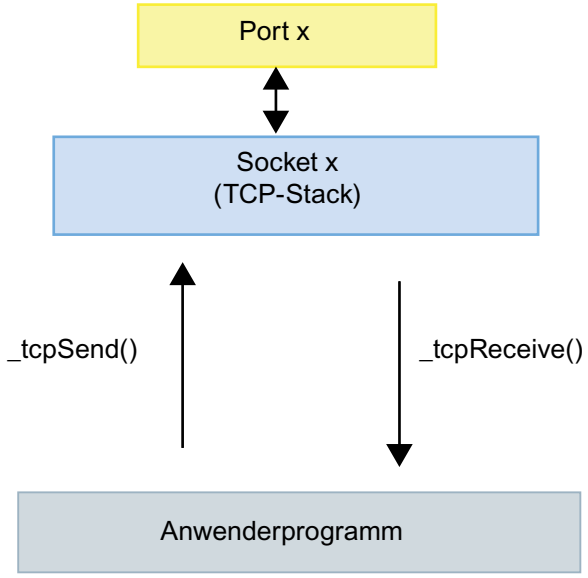


Bild 5-2 Datenaustausch zwischen Anwenderprogramm und TCP-Stack

Datenpakete

Bei der TCP-Kommunikation werden die Sendedaten vom TCP-Stack in beliebig große Teilpakete aufgeteilt (max. 1456 Bytes pro Paket). D. h. auf der Empfängerseite können unterschiedliche Datenpaketgrößen ankommen.

Auf Empfängerseite sind folgenden Szenarien möglich:

- Teilpakete:
  empfangenes Datenpaket < gesendetes Datenpaket
- Mehrere Pakete zu einem großen Datenpaket zusammengefasst:
  empfangenes Datenpaket > gesendetes Datenpaket

Die Reihenfolge der Daten bleibt dabei erhalten. Der Anwender muss in seinem Anwenderprogramm dafür sorgen, dass diese Teilpakete wieder zu der Länge des gesendeten Datenpakets zusammengebaut werden.

Bei der TCP-Kommunikation werden die empfangenen Nutzdaten im TCP-Stack gepuffert. Dort müssen die Nutzdaten per Applikation ausgelesen werden. Die Größe der gepufferten Nutzdaten pro Verbindung ist von der Steuerung abhängig.

Tabelle 5- 1 Nutzdaten und Steuerung

Steuerung	Puffergröße in Bytes
SIMOTION	4096
SIMATIC	8192

Die gepufferten Nutzdaten müssen zeitnah vom Anwenderprogramm abgeholt werden. Ist das nicht der Fall, können keine weiteren Daten mehr empfangen werden. An der Stelle werden keine Nutzdaten verworfen, da TCP eine Flusssteuerung beinhaltet. Der Kommunikationspartner sendet in dem Zustand keine weiteren Daten mehr und signalisiert dies seiner Anwendung.

5.3.2 SIMOTION Systemfunktionen für TCP-Kommunikation

5.3.2.1 Übersicht SIMOTION Systemfunktionen

Maximale Anzahl der möglichen TCP-Verbindungen

Beispielhaft finden Sie in der folgenden Tabelle die Anzahl der möglichen Kommunikationsverbindungen einer SIMOTION-CPU als Client. Die Werte beziehen sich auf ein lokales Netzwerk ohne weitere Fremdlast und eine SIMOTION D435 als Server.

Tabelle 5- 2 Kommunikationsverbindungen in Abhängigkeit der SIMOTION-CPU

SIMOTION-CPU (Client)	Anzahl Kommunikationsverbindungen
C240	45
D410	45
D435	75
P3x0	40

Funktionsaufruf

Die SIMOTION Systemfunktionen für TCP-Kommunikation dürfen nur in der BackgroundTask oder in einer MotionTask aufgerufen werden. Dabei ist in der BackgroundTask zu beachten, dass die Systemfunktionen asynchron abgearbeitet werden (nextCommand=IMMEDIATELY). In der MotionTask können die Systemfunktionen synchron ausgeführt werden (nextCommand=WHEN_COMMAND_DONE).

Hinweis

Um eine sichere zyklische Kommunikation zu gewährleisten, sollten Sie auf Standardmechanismen wie z. B. unter PROFIBUS DP oder PROFINET IRT zurückgreifen.

Siehe auch

Systemfunktion _tcpOpenServer() (Seite 159)
Systemfunktion _tcpOpenClient() (Seite 160)
Systemfunktion _tcpSend() (Seite 161)
Systemfunktion _tcpReceive() (Seite 161)
Systemfunktion _tcpCloseConnection() (Seite 162)
Systemfunktion _tcpCloseServer() (Seite 163)

5.3.2.2 Systemfunktion _tcpOpenServer()

Übersicht

Um einen passiven Verbindungsaufbau zu starten, wird diese Systemfunktion verwendet.

Tabelle 5- 3 Aufrufbeispiel

```
sRetValueTcpOpenServer := _tcpOpenServer(  //StructRetTcpOpenServer
   port        := 3456                     //UINT, 1024-65535
  ,backLog     := 5                        //DINT
  ,nextCommand := IMMEDIATELY              //EnumTcpNextCommandMode
);
```

Zur Parametrierung der Funktion wird für den Parameter **port** der lokal vergebene SIMOTION-Port übergeben.

Als weiteren Parameter wird für **backLog** angegeben, wie viele parallele Verbindungsanforderungen maximal für diesen Port von anderen Steuerungen her zugelassen werden sollen.

Auch für diese Funktion wird mit dem Parameter **nextCommand** das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar und beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

An das Anwenderprogramm wird bei Aufruf der Funktion _tcpOpenServer() eine Struktur zurückgegeben, die folgende Parameter enthält.

Über den Parameter **functionResult** kann der Status des Verbindungsaufbaus abgefragt werden.

Der Parameter **connectionId** dient als (Eingangs-)Parameter für den Aufruf der Funktionen _tcpSend() und _tcpReceive() und ordnet den vorgenannten Funktionen eindeutig eine TCP-Verbindung zu. In den oben genannten Aufruf-Beispielen wird auf diesen Rückgabewert Bezug genommen.

Der Parameter **clientAddress** liefert die IP-Adresse des Clients, von der aus die Verbindung aktiviert wird, in Form eines Arrays.

Im Parameter **clientPort** wird die als lokale Portnummer bezeichnete Portnummer des Clients angegeben.

Die Portnummer liegt im Bereich 1024 bis 65535.

5.3.2.3 Systemfunktion _tcpOpenClient()

Übersicht

Um aktiv eine Verbindung aufzubauen, wird die Systemfunktion _tcpOpenClient() verwendet.

Tabelle 5- 4 Aufrufbeispiel

```
sRetValueTcpOpenClient :=_TCPOpenClient( // StructRetTcpOpenClient
   port           := 3456                // UINT, 1024-65535
  ,serverAddress  := au8ServerAddress    // ARRAY [0...3] OF UINT
  ,serverPort     := 3456                // UINT, 1024-65535
  ,nextCommand    := IMMEDIATELY         // EnumTcpNextCommandMode
  );
```

Beim Aufruf wird der Funktion für den Parameter **port** der lokal vergebene SIMOTION-Port übergeben.

Der Parameter **serverAddress** ist die IP-Adresse des Kommunikationspartners, die in einem Array übergeben wird.

Im Parameter **serverPort** wird der Funktion die als lokale Portnummer bezeichnete Portnummer übergeben.

Die Portnummer liegt im Bereich 1024 bis 65535.

Mit dem Parameter **nextCommand** wird das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar und beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

An das Anwenderprogramm wird bei Aufruf der Funktion _tcpOpenClient() eine Struktur zurückgegeben, die folgende Parameter enthält.

Über den Parameter **functionResult** kann der Status des Verbindungsaufbaus abgefragt werden.

Der Parameter **connectionId** dient als (Eingangs-)Parameter für den Aufruf der Funktionen _tcpSend(), _tcpRecieve() und _tcpCloseConnection() und ordnet den vorgenannten Funktionen eindeutig eine TCP -Verbindung zu.

5.3.2.4 Systemfunktion _tcpSend()

Übersicht

Für das Senden von Daten wird die Systemfunktion _tcpSend() verwendet.

Tabelle 5- 5 Aufrufbeispiel

```
i32RetValue := _tcpSend(                                  // DINT
   connectionId    := sRetValueTcpOpenClient.ConnectionId  // DINT
  ,nextCommand     := IMMEDIATELY                          // EnumTcpNextCommandMode
  ,dataLength      := 4096                                 // UINT, 0-4096
  ,data            := ab8SendData                          // ARRAY [0..4095] OF BYTE
 );
```

Für den Parameter **connectionId** wird der Rückgabewert **connectionId** der Funktionen _tcpOpenClient() oder _tcpOpenServer() übergeben - je nachdem, ob der funktionsausführende Kommunikationsteilnehmer Server oder Client ist.

Für diese Funktion wird ebenfalls mit dem Parameter **nextCommand** das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar und beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

Im Parameter **dataLength** wird der Funktion mitgeteilt, wie viele Nutzdaten in Bytes übertragen werden sollen (max. 4096 Bytes pro Funktionsaufruf).

Mit dem Parameter **data** wird angegeben, in welchem Nutzdatenbereich die Sendedaten liegen, die mit der Funktion übertragen werden sollen.

Der Rückgabewert der Funktion an das Anwenderprogramm hat den Datentyp DINT. Über unterschiedliche Rückgabewerte wird auf Probleme bei der Ausführung der Funktion hingewiesen. Ebenso wird zurückgemeldet, wenn die Daten erfolgreich gesendet wurden. Negative Werte in **functionResult** weisen auf einen Fehler bei der Datenübertragung hin. In diesem Fall muss die Verbindung durch Aufruf von _tcpCloseConnection() abgebaut werden.

5.3.2.5 Systemfunktion _tcpReceive()

Übersicht

Um Daten zu empfangen, wird die Systemfunktion _tcpReceive() verwendet.

Tabelle 5- 6 Aufrufbeispiel

```
sRetValueTcpReceive := _tcpReceive(                       // StructRetTcpReceive
   connectionId     := sRetValueTcpOpenClient.connectionId  // DINT
  ,nextCommand      := IMMEDIATELY                          // EnumNextCommandMode
  ,receiveVariable  := ab8ReceiveData                       // ARRAY [0..4095] OF BYTE
  );
```

Für den Parameter **connectionId** wird der Rückgabewert **connectionId** der Funktionen _tcpOpenClient() oder _tcpOpenServer() übergeben – je nachdem, ob der funktionsausführende Kommunikationsteilnehmer Server oder Client ist.

Für _tcpReceive() wird ebenfalls mit dem Parameter **nextCommand** das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Es gibt zwei Einstellmöglichkeiten: IMMEDIATELY und WHEN_COMMAND_DONE. Beim ersten Wert wird unmittelbar bzw. beim zweiten Wert erst nach Beendigung des Befehls weitergeschaltet.

Pro Funktionsaufruf können bis zu 4096 Bytes Empfangsdaten aus dem TCP-Stack ausgelesen werden. Dabei ist zu beachten, dass die empfangenen Pakete in nicht vorhersagbare Größe vorliegen. Auf der Empfängerseite muss dafür Sorge getragen werden, dass alle Nutzdaten vor der Auswertung und weiteren Verarbeitung vorhanden sind. Hierzu sollte der Parameter **nextCommand** auf IMMEDIATELY gesetzt werden.

Mit dem Parameter **receiveVariable** wird der Funktion mitgeteilt, in welchem Nutzdatenbereich die Empfangsdaten abgelegt werden sollen. Die empfangenen Daten stehen im Parameter **receiveVariable** in der Länge **dataLength** zur Verfügung, wenn im functionresult der Rückgabewert = 16#0 ist. **ReceiveVariable** wird beim nächsten Aufruf der Funktion mit neuen Daten in der Länge **dataLength** überschrieben.

An das Anwenderprogramm wird bei Aufruf der Funktion _tcpReceive() eine Struktur zurückgegeben, die folgende Parameter enthält. Über den Parameter **functionResult** kann der Status des Empfangens abgefragt werden. Parameter **dataLength** meldet die Anzahl der empfangenen Nutzdatenbytes nach erfolgreichem Aufruf der Funktion _tcpReceive() zurück.

Negative Werte in **functionResult** weisen auf einen Fehler bei der Datenübertragung hin. In diesem Fall muss die Verbindung durch Aufruf von _tcpCloseConnection() abgebaut werden.

5.3.2.6 Systemfunktion _tcpCloseConnection()

Übersicht

Um eine Verbindung zu schließen, die vom funktionsausführenden Kommunikationsteilnehmer aktiv aufgebaut wurde (Client), wird die Funktion _tcpCloseConnection() verwendet.

Tabelle 5- 7 Aufrufbeispiel

```
sRetValueTcpCloseConnection := _tcpCloseConnection(      // DINT
   connectionId := sRetValueTcpOpenClient.ConnectionId   // DINT
  );
```

Am Parameter **connectionId** wird der Rückgabewert der Funktion _tcpOpenClient() übergeben, um eindeutig festzulegen, welche Verbindung abgebaut werden soll.

Der Rückgabewert der Funktion an das Anwenderprogramm hat den Datentyp DINT und weist auf Probleme bei der Ausführung der Funktion hin bzw. meldet zurück, wenn die Verbindung erfolgreich abgebaut wurde.

5.3.2.7 Systemfunktion _tcpCloseServer()

Übersicht

Um eine Verbindung zu schließen, die vom funktionsausführenden Kommunikationsteilnehmer passiv aufgebaut wurde (Server), wird die Funktion _tcpCloseServer() verwendet.

Tabelle 5- 8 Aufrufbeispiel

```
sRetValueTcpCloseServer := _tcpCloseServer(port := 3456); //1024 - 65535
```

Am Parameter **port** wird der Port des Servers übergeben.

Der Rückgabewert der Funktion an das Anwenderprogramm hat den Datentyp DINT und weist auf Fehler in der Parametrierung der Funktion hin bzw. meldet zurück, wenn der Port erfolgreich geschlossen wurde.

5.3.3 SIMATIC Kommunikationsbausteine onboard Ethernet-Schnittstelle

5.3.3.1 Übersicht SIMATIC Kommunikationsbausteine

Übersicht

Für die TCP–Kommunikation mit einer SIMATIC und deren onboard-Ethernet-Schnittstellen stehen folgende Funktionsbausteine zur Verfügung.

- TSEND (FB63)
- TRCV (FB64)
- TCON (FB65)
- TDISCON (FB66)

Um die Verbindung aufzubauen, benötigt der FB65 TCON die Datenstruktur UDT65 TCON_PAR. Als Rückgabewert bekommt die Datenstruktur eine Verbindungs-ID, welche beim Aufruf der Kommunikationsbausteine übergeben werden muss. Zum Abbauen der Verbindung wird der Funktionsbaustein FB66 TDISCON verwendet. Das Senden und Empfangen der Nutzdaten wird mittels der Bausteine FB63 TSEND und FB64 TRCV durchgeführt.

Maximale Anzahl der Verbindungen

Tabelle 5- 9 Steuerungstypen und max. Anzahl der Verbindungen

SIMATIC	
CPU 315(F)-2 PN/DP	8
CPU 317(F)-2 PN/DP	8
CPU 319(F)-2 PN/DP	32
CPU 414-3 PN/DP	30
CPU 416(F)-3 PN/DP	62
IM151-8(F) PN/DP CPU	8

Siehe auch

Übersicht SIMATIC Kommunikationsbausteine (Seite 182)

5.3.3.2 Aufbau und Parametrierung UDT65

Übersicht

Bei der Kommunikation mit einer SIMATIC Steuerung und integrierter Ethernet-Schnittstelle entfällt die Verbindungsprojektierung in NetPro. Stattdessen wird die Verbindungsprojektierung mithilfe der Datenstruktur UDT65 TCON_PAR im Anwenderprogramm vorgenommen (siehe Bild unten).

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	block_length	WORD	W#16#40	Länge 64 Byte
+2.0	id	WORD	W#16#0	xxx Verbindungs ID
+4.0	connection_type	BYTE	B#16#1	
+5.0	active_est	BOOL	FALSE	
+6.0	local_device_id	BYTE	B#16#2	
+7.0	local_tsap_id_len	BYTE	B#16#2	
+8.0	rem_subnet_id_len	BYTE	B#16#0	
+9.0	rem_staddr_len	BYTE	B#16#0	xxx 0: unspezifizierte Verb. 4: spezifizierte Verb.
+10.0	rem_tsap_id_len	BYTE	B#16#0	
+11.0	next_staddr_len	BYTE	B#16#0	
+12.0	local_tsap_id	ARRAY[1..16]	B#16#0	xxx lokaler Port
*1.0		BYTE		
+28.0	rem_subnet_id	ARRAY[1..6]	B#16#0	
*1.0		BYTE		
+34.0	rem_staddr	ARRAY[1..6]	B#16#0	xxx Remote IP-Adr.
*1.0		BYTE		
+40.0	rem_tsap_id	ARRAY[1..16]	B#16#0	
*1.0		BYTE		
+56.0	next_staddr	ARRAY[1..6]	B#16#9	
*1.0		BYTE		
+62.0	spare	WORD	W#16#0	
=64.0		END_STRUCT		

Bild 5-3 Deklarationssicht des UDT65 TCON_PAR im KOP-/AWL-/FUP-Editor

Der Aufbau und die Parametrierung des UDT65 sollen im Folgenden näher erläutert werden.

Bei der Parametrierung des UDT65 bzw. des sich daraus ableitenden Datenbausteins unterscheidet man prinzipiell zwei Fälle:

1. Der Verbindungsaufbau geht aktiv von der S7 Station aus (Client).
2. Die S7 Station wartet passiv auf einen Verbindungsaufbau vom Kommunikationspartner (Server).

Die wichtigsten Parameter finden Sie in der tabellarischen Übersicht (nach Client und Server unterschieden). Nicht erwähnte Parameter werden in der Regel auf ihren voreingestellten Werten oder auf 0 belassen. Für weitere Informationen benutzen Sie die Online-Hilfe des Datenbausteins UDT65.

Tabelle 5- 10 Datenbaustein UDT65, Parametrierung bei TCP bzw. UDP

Parameter	Beschreibung	Wert für TCP	Wert für UDP
block_length	Länge eines Parametrierblocks	64 hex	64 hex
id	Für jede Verbindung muss eine eigene Nummer über den Parameter id (Wertebereich 1-0FFF hex) vergeben werden. Diese Referenz wird für die Parametrierung der Funktionsbausteine TCON, TSEND, TRCV und TDISCON benötigt	Den Wert dieses Parameters müssen Sie im jeweiligen Baustein bei ID angeben.	Den Wert dieses Parameters müssen Sie im jeweiligen Baustein bei ID angeben.
connection_type	Verbindungstyp	11 hex	13 hex
local_device_id	Der Parameter local_device_id ist abhängig vom CPU-Typ. Die Werte dazu entnehmen Sie der STEP 7 Online-Hilfe.	-	-
local_tsap_id_len	verwendete Länge des Parameters local_tsap_id	2 hex	2 hex
rem_staddr_len	Länge der Adresse des remoten Verbindungsendpunkts	4 hex	0, nicht verwendet
active_est	**1:** aktiver Verbindungsaufbau	TRUE	FALSE
	2: passiver Verbindungsaufbau	FALSE	FALSE

Parameter	Beschreibung	Wert für TCP	Wert für UDP
rem_staddr	zu 1 und 2, IP-Adresse des Kommunikationspartners z . B. 192.168.0.1 (SIMOTION Gerät).	rem_staddr[1] = B#16#C0 (192), rem_staddr[2] = B#16#A8 (168), rem_staddr[3] = B#16#00 (0), rem_staddr[4] = B#16#01 (1), rem_staddr[5-6]= B#16#00 (reserviert)	0, nicht verwendet
rem_tsap_id	zu 1, remote Port-Nr. Werte gelten bei connection_type 11hex.	B#16#11: rem_tsap_id[1] = high byte der Port-Nr. in hexadezimaler Darstellung, rem_tsap_id[2] = low byte der Port-Nr. in hexadezimaler Darstellung, rem_tsap_id[3-16] = B#16#00	0, nicht verwendet
	Zu 2,	0, nicht verwendet	0, nicht verwendet

Im folgenden Kapitel finden Sie Beschreibungen der Kommunikationsbausteine.

Siehe auch

Übersicht SIMATIC Kommunikationsbausteine (Seite 182)

5.3.3.3 Beschreibung der Kommunikationsbausteine

Funktionsbausteine für den Verbindungsaufbau

Es folgt die Beschreibung der Funktionsbausteine, mit denen ein Verbindungsaufbau programmiert werden kann.

TCON (FB65)

Tabelle 5- 11 Aufrufbeispiel

```
CALL "TCON" , DB66
    REQ       :=M1.0
    ID        :=W#16#1
    DONE      :=M2.0
```

```
    BUSY      :=M3.0
    ERROR     :=M4.0
    STATUS    :=MW100
    CONNECT   :=P#DB1.DBX0.0 BYTE 64
```

Sollen Daten auf einer S7 Station mit integrierter Ethernet-Schnittstelle empfangen bzw. gesendet werden, so muss zuerst über den Funktionsbaustein TCON (FB65) eine Verbindung zwischen der S7 Station und dem Kommunikationspartner aufgebaut werden.

Über den Parameter **REQ** wird der Verbindungsaufbau gesteuert. Wird der Parameter auf den Wert 1 gesetzt und somit eine Flanke erzeugt, so werden die Daten (Verbindungsbeschreibung) aus dem unter **CONNECT** angegebenen Bereich dem Funktionsbaustein übergeben, um die Verbindung aufzubauen.

Durch den Parameter **ID** wird eine Referenz zur gewünschten Verbindung, die aufgebaut werden soll, hergestellt.

Über die Parameter **DONE**, **BUSY** und **ERROR** kann der Status der Bearbeitung des Funktionsbausteins abgefragt werden. Zusätzlich zur Information, dass ein Fehler aufgetreten ist (ERROR = 1), erhält der Benutzer noch durch den Parameter **STATUS** detaillierte Auskunft über die Art des Fehlers.

Wie bereits oben erwähnt, enthält der Parameter **CONNECT** die Adressen und Länge der Verbindungsbeschreibung. Diese Adresse weist auf einen Datenbausteinbereich hin, dessen Struktur der UDT65 entspricht.

TSEND (FB63)

Tabelle 5- 12 Aufrufbeispiel

```
Call "TSEND" , DB63
    REQ         :=M5.0
    ID          :=W#16#1
    LEN         :=10
    DONE        :=M6.0
    BUSY        :=M7.0
    ERROR       :=M8.0
    STATUS      :=MW200
    DATA        :=DB10.DBBO
```

Ist eine Kommunikationsverbindung aufgebaut, so können diese Daten gesendet werden. Dies geschieht durch den Aufruf des Funktionsbausteins TSEND (FB63).

Das Senden wird mit einer steigenden Flanke am Parameter **REQ** aktiviert. Beim erstmaligen Aufruf werden die Daten aus dem mit Parameter **DATA** angegebenen Bereich an den Funktionsbaustein übergeben.

Über den Parameter **ID** wird die Kommunikationsverbindung referenziert, über die die Daten versendet werden sollen. Mit dem Parameter LEN wird die Länge der zu versendenden Daten in Bytes angegeben.

Auch hier geben die Parameter **DONE**, **BUSY** und **ERROR** den Bearbeitungsstatus des Funktionsbausteins an. Zusätzlich zur Information, dass ein Fehler aufgetreten ist (ERROR = 1), erhält der Benutzer durch den Parameter **STATUS** noch detaillierte Auskunft über die Art des Fehlers.

Der Parameter **DATA** enthält, wie bereits oben angegeben, die Adresse und Länge des Sendebereichs.

TRCV (FB64)

Tabelle 5- 13 Aufrufbeispiel

```
CALL "TRCV" , DB64
     EN_R       :=M8.0
     ID         :=W#16#1
     LEN        :=10
     NDR        :=M9.0
     BUSY       :=10.0
     ERROR      :=11.0
     STATUS     :=MW300
     RCVD_LEN   :=MW310
     DATA       :=DB20.DBB0
```

Über eine aufgebaute Verbindung können auch Daten mit dem Funktionsbaustein TRCV (FB64) empfangen werden.

Mit dem Parameter **EN_R** wird das Empfangen gesteuert. D. h., wird der Parameter **EN_R** mit dem Wert 1 beschrieben, so können Daten empfangen werden.

Über ID wird eine bestimmte Kommunikationsverbindung ausgewählt, über die die Daten empfangen werden sollen.

Für den Parameter **LEN** existieren zwei prinzipielle Parametrierungen. Wird der Parameter mit dem Wert 0 beschrieben, so wird implizit die Länge der erwarteten Empfangsdaten über einen ANY-Pointer am Bausteineingang **DATA** angegeben. Sobald Daten empfangen wurden, werden die Daten im Empfangspuffer zur Verfügung gestellt und dies wird über den Parameter **NDR** gemeldet. Die Länge der empfangenen Daten kann dem Parameter **RCVD_LEN** entnommen werden und sie kann auch kleiner sein als die im Parameter **DATA** hinterlegte Größe. Wird der Parameter **LEN** mit einem Wert verschieden von 0 parametriert, so werden die empfangenen Daten im Empfangspuffer zwischengespeichert und stehen erst dann zur Verfügung, wenn die projektierte Länge erreicht wird. Dass die Daten vollständig empfangen wurden, wird ebenfalls über den Parameter **NDR** gemeldet.

Der Parameter **NDR** meldet das teilweise oder komplette Empfangen von Daten.

Für das Empfangen geben die Parameter **DONE**, **BUSY** und **ERROR** den Bearbeitungsstatus des Funktionsbausteins wieder. Zusätzlich zur Information, dass ein Fehler aufgetreten ist (ERROR = 1), erhält der Benutzer durch den Parameter **STATUS** noch detaillierte Auskunft über die Art des Fehlers.

Die Bedeutung des Parameters **RCVD_LEN** wurde oben schon erwähnt. Für den Fall, dass Parameter **LEN** mit dem Wert 0 belegt wurde, wird im Parameter **RCVD_LEN** angegeben, wie viele Daten in dem zuletzt empfangenen Datenblock enthalten waren. Wurde im Parameter **LEN** ein von 0 verschiedener Wert parametriert, so steht in **RCVD_LEN** derselbe Wert.

Der Parameter **DATA** enthält die Adresse und Länge des Sendebereichs. Dort können die empfangenen Daten zur weiteren Verarbeitung entnommen werden.

FDISCON (FB66)

Tabelle 5- 14 Aufrufbeispiel

```
CALL    "TDISCON" , DB66
    REQ     :=M12.0
    ID      :=W#16#1
    DONE    :=M13.0
    BUSY    :=M14.0
    ERROR   :=M15.0
    STATUS  :=MW400
```

Der Funktionsbaustein TDISCON (FB66)wird benutzt, um eine bestehende Verbindung abzubauen. Um die Verbindung abzubauen, wird der Eingangsparameter **REQ** auf den Wert 1 gesetzt. Der Anstoß zum Abbau der Verbindung erfolgt also durch die steigende Flanke.

Über den Parameter **ID** wird dem Funktionsbaustein mitgeteilt, welche Verbindung abgebaut werden soll. Über diesen Parameter wird eine Referenz zu einer mittels einer Struktur vom Typ **TCON_PAR** definierten und bereits aufgebauten Verbindung angegeben.

Über die Parameter **DONE**, **BUSY** und **ERROR** kann der Status der Bearbeitung des Funktionsbausteins abgefragt werden. Zusätzlich zur Information, dass ein Fehler aufgetreten ist (ERROR = 1), erhält der Benutzer noch durch den Parameter **STATUS** detaillierte Auskunft über die Art des Fehlers.

5.3.4 SIMATIC Kommunikationsbausteine für Ethernet-CP

5.3.4.1 Übersicht SIMATIC Kommunikationsbaustein

Übersicht

Es stehen folgende Kommunikationsbausteine zur Verfügung:

- AG_SEND (FC5)
- AG_RECV (FC6)
- AG_LSEND (FC50)
- AG_LRECV (FC60)
- AG_SSEND (FC53)
- AG_SRECV (FC63)

Im Folgenden werden nur die SIMATIC Kommunikationsbausteine AG_SEND und AG_RECV beschrieben.

Weitere Informationen finden Sie dazu im Programmierhandbuch *SIMATIC NET Funktionen (FC) und Funktionsbausteine (FB) für SIMATIC NET S7-CPs*.

Hinweis

Für die **CP 300** beträgt die maximale Sende- und Empfangslänge pro Funktionsaufruf (AG_SEND und AG_RECV) 8192 Bytes.

Für die **CP 400** können ebenfalls die Funktionen AG_SEND und AG_RECV verwendet werden. Allerdings ist hier die übertragbare Datenlänge generell pro Auftrag auf <=240 Bytes beschränkt!

5.3.4.2 Projektierung der Ethernet-CP

Vorgehensweise

1. Um die Kommunikations-Verbindung anzulegen und zu projektieren, muss in NetPro die Verbindungstabelle der S7-Station dargestellt sein. Hierzu wird die CPU innerhalb der S7-Station markiert. Im unteren Arbeitsbereich von NetPro wird nun die Verbindungstabelle angezeigt. Für das SIMOTION Gerät kann in NetPro keine Verbindungstabelle angezeigt werden.

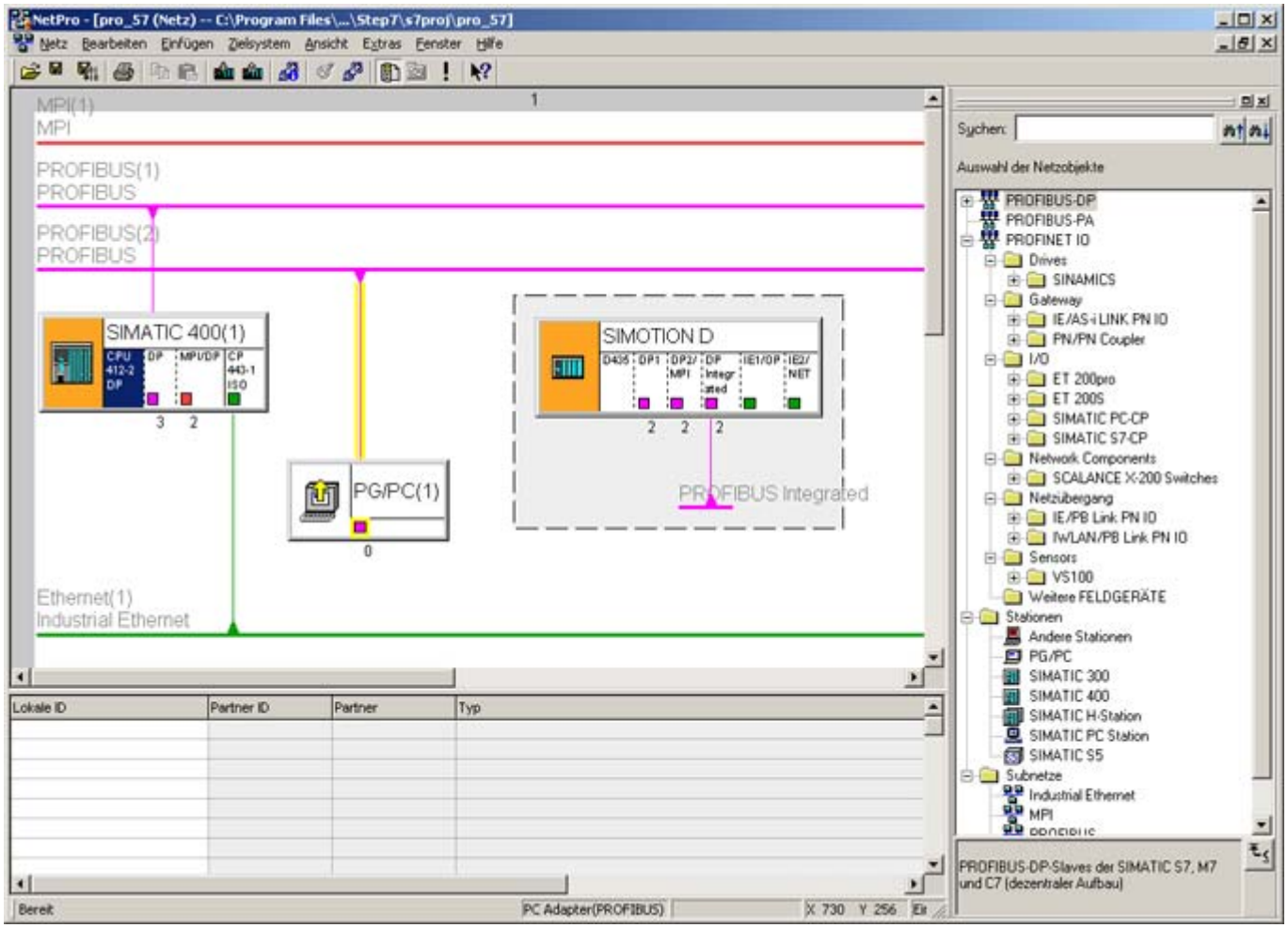


Bild 5-4 Markierte S7 CPU und die dazugehörige Verbindungstabelle

2. Durch Doppelklick auf eine leere Zeile in der Verbindungstabelle wird der Dialog **Neue Verbindung einfügen** zum Einfügen einer neuen Kommunikations-Verbindung geöffnet. Der Dialog ist für TCP- und UDP-Verbindung gleich. Er kann bei markierter CPU auch über das Kontextmenü **Einfügen > Neue Verbindung…** oder durch Klicken auf den Button in der Menüleiste erreicht werden.

3. Im Dialog **Neue Verbindung einfügen** wählen Sie den Verbindungspartner aus.

Hinweis

Bei Kommunikation zwischen einem S7 CP und einem SIMOTION Gerät wählen Sie die Einstellung "(unspezifiziert)".

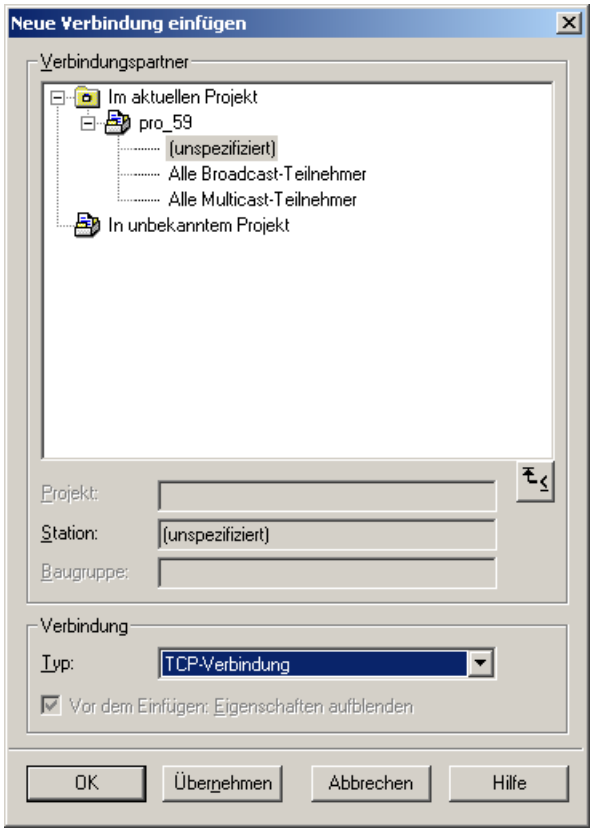


Bild 5-5 Dialog "Neue Verbindung einfügen" mit ausgewählter TCP-/IP-Verbindung

4. Im Feld Verbindung wird der gewünschte Verbindungs-Typ TCP-Verbindung ausgewählt.

Hinweis

Bei UDP-Kommunikation wählen Sie hier UDP-Verbindung.

5. Wird der Dialog **Neue Verbindung einfügen** mit **OK** verlassen, erscheint zunächst eine Hinweismeldung, die darauf verweist, dass auch Verbindungen über Subnetze hinweg möglich sind und eventuell die Routeradressen überprüft werden sollten.
   Nach dem Quittieren dieser Hinweismeldung öffnet sich für eine TCP-Verbindung der Dialog **Eigenschaften**.

Bild 5-6 Dialog Eigenschaften - TCP-Verbindung" - Register "Adressen"

6. Im Register **Adressen** sind die IP-Adresse und auch der Port für den lokalen Kommunikationspartner schon vorbelegt. Für den fernen Kommunikations-Partner sind die Angaben noch zu vervollständigen. In das Feld **IP (DEZ)** ist die IP-Adresse des Kommunikationspartners einzutragen. Im Feld **Port (DEZ)** ist ein Port anzugeben, der beim Kommunikationspartner für diese Verbindung vom Anwender festgelegt wurde. Für den Port auf S7-Seite sind Randbedingungen einzuhalten, d. h., auf S7-Seite muss ein Port zwischen 2000 und 5000 ausgewählt werden.

Hinweis

SIMOTION unterstützt Port 1024 bis 65535.

7. Im Register **Allgemein** können im Feld **Bausteinparameter** die Parameter **ID** und **LADDR** für die S7-Kommunikationsbausteine entnommen werden. Mit ID wird der Kommunikationsverbindung eine eindeutige Referenz zugewiesen. Zudem wird als **LADDR** die Adresse des CP angegeben.
   Über die Anwahl der Check-Box Aktiver Verbindungsaufbau kann festgelegt werden, ob der Verbindungsaufbau von der S7-Station aus erfolgen soll. Wenn auf S7-Seite ein aktiver Verbindungsaufbau angewählt ist, so muss der Kommunikationspartner als Server projektiert werden.
   Wird auf S7-Seite jedoch kein aktiver Verbindungsaufbau angewählt, so muss der Kommunikationspartner als Client projektiert werden.

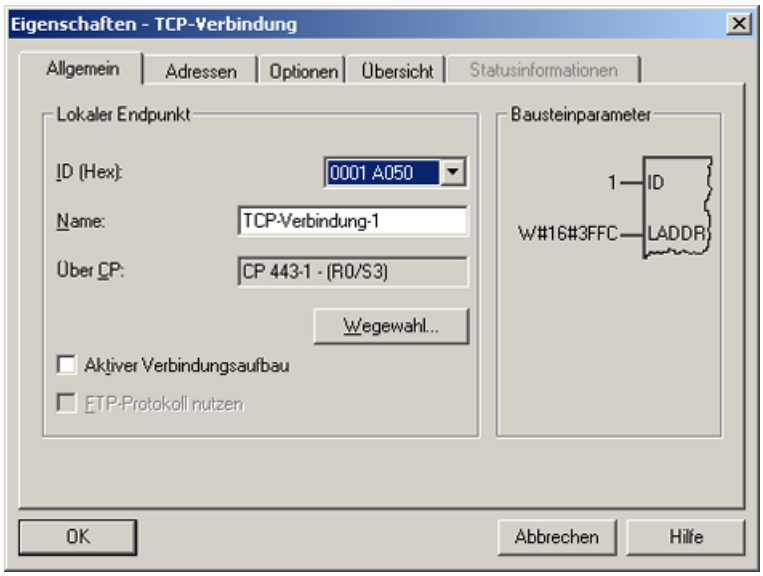


Bild 5-7 Dialog "Eigenschaften - TCP-Verbindung" - Register "Allgemein"

8. Wird der Dialog Eigenschaften nun mit **OK** verlassen, so muss noch der Dialog **Neue Verbindung einfügen** durch Klicken des Buttons **Schließen** geschlossen werden, um die Verbindungs-Projektierung abzuschließen. Möchte man weitere Verbindungen projektieren, so ist dies durch Anwahl des gewünschten Verbindungs-Typs und anschließendem Klicken auf den Button **Übernehmen** möglich.
   Nach erfolgter Projektierung der Kommunikations-Verbindung stehen die Parameter für den Verbindungsaufbau mit einem Kommunikationspartner fest.

5.3.4.3 Beschreibung der Kommunikationsbausteine

Übersicht

Im Folgenden finden Sie eine Beschreibung der Kommunikationsbausteine, mit denen ein Verbindungsaufbau programmiert werden kann.

AG_SEND (FC5)

Tabelle 5- 15 Aufrufbeispiel

```
CALL "AG_Send"
 Act :=M0.0
 ID :=1
 LADDR :=W#16#3FFC
 SEND :=P#DB100.DBX0.0 BYTE 1000
 LEN :=1000
 DONE :=M0.1
 ERROR :=M0.2
 STATUS :=MW10
```

Die Parameter **ID** und **LADDR** werden beim Anlegen der Verbindung in NetPro angezeigt und müssen für diese Verbindung beim Aufruf der Funktionen übergeben werden. Mit einer positiven Flanke am Eingang **ACT** wird das Senden angestoßen. Der Parameter **SEND** definiert die Sendedaten und über **LEN** wird die zu sendende Länge festgelegt. Die Parameter **DONE**, **ERROR** und **STATUS** dienen der Diagnose bzw. liefern den Status zurück, den der Sendeauftrag hat.

AG_RECV (FC6)

Tabelle 5- 16 Aufrufbeispiel

```
Call "AG_RECV"
 ID :=1
 LADDR :=W#16#3FFC
 RECV :=P#DB110.DBX0.0 BYTE 1000
 NDR :=M2.0
 ERROR :=M2.1
 STATUS :=MW14
 LEN :=MW16
```

Beim Anlegen der Verbindung in NetPro werden die Parameter **ID** und **LADDR** von NetPro festgelegt und müssen für diese Verbindung beim Aufruf der Funktionen übergeben werden.

Am Parameter **RECV** wird ein Pointer übergeben, welcher auf den Datenbereich zeigt, in dem die empfangenen Daten abgelegt werden. Mit **TRUE** am Ausgang **NDR** wird dem Anwender mitgeteilt, wenn neue Daten empfangen wurden. Der Parameter **LEN** gibt an, wie viele Bytes ausgelesen wurden. Über die Parameter **ERROR** und **STATUS** wird eine Diagnose bzw. der Status zurück geliefert, den der Empfangsaufruf hat.

Hinweis

Bei der TCP-Kommunikation wird der Ausgang NDR erst gesetzt, wenn der Datenbereich für die empfangenen Daten voll ist.

5.4 UDP-Kommunikation

5.4.1 Übersicht UDP-Kommunikation

Kommunikation mit UDP (User Datagram Protocol)

UDP stellt ein Verfahren zur Verfügung, um aus dem Anwenderprogramm mit einem Minimum von Protokoll-Mechanismus Daten über Ethernet zu senden und zu empfangen. Eine Rückinformation bezüglich der übertragenen Daten findet bei Kommunikation über UDP nicht statt. Die Kommunikation findet sendeseitig und empfangsseitig über Ports statt. Im Gegensatz zu TCP müssen Sie keinen Verbindungsaufbau bzw. –abbau programmieren.

Prinzip der UDP-Kommunikation

- Für den Empfang adressieren Sie im Befehl den Port, den Sie auf Ihrer Komponente für den Kommunikationsauftrag nutzen möchten.
- Sie geben beim Senden von Daten die IP-Adresse und Portnummer des Zielsystems, sowie die Port-Nummer der lokalen Steuerung an.
- Sie können angeben, ob der Port auf Ihrer Seite nach der Ausführung des Kommunikationsauftrages reserviert bleiben soll.
  UDP ist kein gesichertes Modell. Es können deshalb Daten bei der Übertragung verloren gehen, falls die Daten nicht rechtzeitig aus dem Puffer gelesen werden. Eine gesicherte Datenübertragung müssen Sie über Ihre Applikation programmieren, z. B. durch Quittieren des Datenempfangs.
- Für das Senden sind folgenden Daten mindestens notwendig:
  - IP-Adresse des Kommunikationspartners
  - die "eigene" Port-Nummer
  - die Port-Nummer des Kommunikationspartners
- UDP ist kein sicheres Übertragungsprotokoll. Eine Rückmeldung über den Erfolg der Datenübertragung müssen Sie im Anwenderprogramm selbst programmieren.

Datenaustausch zwischen Anwenderprogramm und UDP-Stack

Folgendes Bild zeigt das UDP-Kommunikationsmodell auf SIMOTION-Seite.

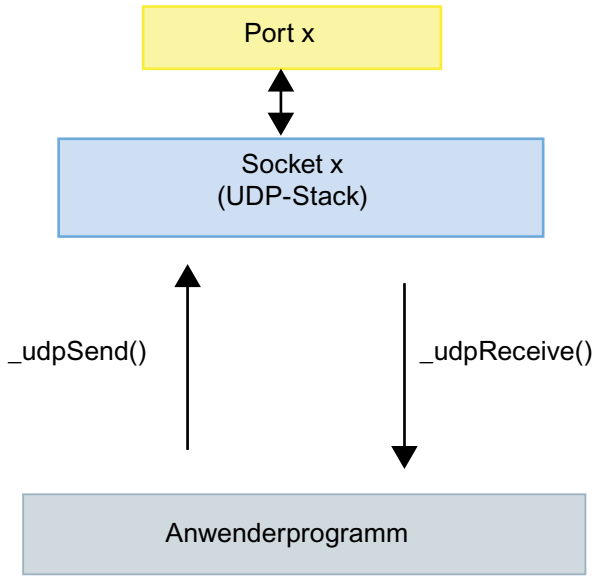


Bild 5-8 Datenaustausch zwischen Anwenderprogramm und UDP-Stack

Bei der UDP-Kommunikation werden die empfangenen Nutzdaten im UDP-Stack gepuffert. Dort müssen die Nutzdaten per Applikation ausgelesen werden. Die Größe der gepufferten Nutzdaten pro Verbindung ist von der Steuerung abhängig. Wird der Empfangspuffer nicht rechtzeitig ausgelesen, gehen die empfangenen UDP-Telegramme verloren.

Tabelle 5- 17 Nutzdaten und Steuerung

Steuerung	Gepufferte Nutzdaten in Bytes
SIMOTION	1470 (ab SIMOTION V4.1 SP4) 1400 (bis SIMOTION V4.1 SP4)
SIMATIC	bis zu 2048 (abhängig von CPU und Kommunikation über Ethernet-CP oder integrierter Ethernet-Schnittstelle)

5.4.2 SIMOTION Systemfunktionen für UDP-Kommunikation

5.4.2.1 Übersicht SIMOTION Systemfunktionen

Funktionsaufruf

Die UDP Systemfunktionen der SIMOTION dürfen nur in der BackgroundTask oder in einer MotionTask aufgerufen werden.

Das Versenden von Daten läuft über _udpSend(). Sollen Daten auf SIMOTION-Seite empfangen werden, so wird die Funktion _udpReceive() verwendet. In den folgenden Kapiteln werden die Funktionen beschrieben.

Hinweis

Um eine sichere zyklische Kommunikation zu gewährleisten, sollten Sie auf Standardmechanismen wie z. B. unter PROFIBUS DP oder PROFINET IRT zurückgreifen.

Siehe auch

Systemfunktion _udpSend() (Seite 179)

Systemfunktion _udpReceive() (Seite 180)

Systemfunktion _udpAddMulticastGroupMembership() (Seite 181)

Systemfunktion _udpDropMulticastGroupMembership() (Seite 182)

5.4.2.2 Systemfunktion _udpSend()

Übersicht

Für das Senden von Daten wird die Systemfunktion _udpSend() verwendet.

Tabelle 5- 18 Aufrufbeispiel

```
i32RetValue :=_udpSend(
   sourcePort         := 3456                    //UINT, 1024 - 65535
  ,destinationAddress := au8DestinationAddress   //ARRAY[0..3] OF USINT
  ,destinationPort    := u16DestinationPort      //UINT, 1024 - 65535
  ,communicationMode  := CLOSE_ON_EXIT           //EnumUdpCommunicationMode
  ,dataLength         := u32DataLength           //UDINT
  ,data               := ab8SendData                 //ARRAY[0..1469] OF BYTE
  );
```

Beim Aufruf der Funktion _udpSend() wird für den Parameter **sourcePort** der lokale Port übergeben. Der Parameter **destinationAddress** ist die IP-Adresse, die in einem Array übergeben wird. Die IP-Adresse kann in HW Konfig konfiguriert und ausgelesen werden.

Als **destinationPort** wird der Port des Kommunikationspartners übergeben.

Über **communicationMode** kann der Anwender festlegen, ob die Kommunikations-Ressourcen nach dem Senden frei gegeben (CLOSE_ON_EXIT) oder nicht frei gegeben (DO_NOT_CLOSE_ON_EXIT) werden sollen.

Die Parameter **dataLength** und **data** geben die zu versendende Datenlänge bzw. den Bereich an, wo die Sendedaten abgelegt sind.

Über den Rückgabewert der Funktion kann man den Status des Sendeauftrags überprüfen.

Hinweis

Bis zu SIMOTION 4.1 SP4 war die Länge der Sendedaten (data) auf 1400 Bytes begrenzt.

5.4.2.3 Systemfunktion _udpReceive()

Übersicht

Für das Empfangen von Daten wird die Systemfunktion _udpReceive() aufgerufen.

Tabelle 5- 19 Aufrufbeispiel

```
sRetValueUdpReceive :=_udpReceive(
   port               := 3456                  //UINT, 1024 - 65535
  ,communicationMode  := CLOSE_ON_EXIT         //EnumUdpCommunicationMode
  ,nextCommand        := IMMEDIATELY           //EnumNextCommandMode
  ,receiveVariable    := ab8ReceiveData               //ARRAY[0..1469] OF BYTE
  );
```

Beim Aufruf der Funktion wird für den Parameter **port** der lokale Port übergeben.

Auch hier kann über **communicationMode** vom Anwender festgelegt werden, ob die Kommunikations-Ressourcen nach dem Empfangen frei gegeben (CLOSE_ON_EXIT) oder nicht frei gegeben (DO_NOT_CLOSE_ON_EXIT) werden sollen.

Mit dem Parameter **nextCommand** wird das Verhalten der Funktion in Bezug auf das Weiterschalten beim Aufruf parametriert. Für diesen Parameter gibt es drei Einstellmöglichkeiten: IMMEDIATELY, WHEN_COMMAND_DONE und ABORT_CURRENT_COMMAND. Bei den beiden ersten Werten wird entweder unmittelbar oder eben erst nach Beendigung des Befehls weitergeschaltet. Mit dem dritten Wert wird bei Übergabe derselben Port-Nummer wie beim vorhergehenden Aufruf der Funktion, die aktive Funktion abgebrochen.

Der Parameter **receiveVariable** gibt den Puffer an, in den die Empfangsdaten gelegt werden.

An das Anwenderprogramm wird bei Aufruf der Systemfunktion _udpReceive() eine Struktur zurückgegeben, die folgende Parameter enthält. Im Parameter **functionResult** kann der Status des Aufrufs der Empfangsfunktion abgefragt werden.

Der Parameter **sourceAddress** ist ein Array, das die IP-Adresse enthält. Ebenso enthält der Parameter **sourcePort** der Struktur den lokalen Port.

Im Parameter **dataLength** kann die Anzahl der empfangenen Nutzdatenbytes nach erfolgreichem Aufruf der Systemfunktion _udpReceive() ausgelesen werden.

Hinweis

Bis zu SIMOTION 4.1 SP4 war die Länge der Empfangsdaten (receiveVariable) auf 1400 Bytes begrenzt.

5.4.2.4 Systemfunktion _udpAddMulticastGroupMembership()

Übersicht

Diese Funktion dient dazu, einer Multicastgruppe an einer ausgewählten Ethernet-Schnittstelle beizutreten.

Es können maximal 3 Multicastgruppen an einer Ethernet-Schnittstelle angelegt werden. Diese können alle auf demselben oder auf unterschiedlichen Ports liegen.

```
myRetDINT :=
  _udpAddMulticastGroupMembership(
      multicastIPAddress :=
      ,interfaceIPAddress :=
      ,multicastPort :=
      // ,multicastTTL := 1
      // ,multicastLOOP := 1
      ,nextCommand :=
);
```

Hinweis

Bei einer UDP Multicast-Kommunikation muss der _multicastPort dem sourcePort bei der Funktion _udpSend() entsprechen.

Hinweis

Die UDP Multicast Funktionen (_updAddMulticastGroupMembership(), updDropMulticastGroupMembership()), sind nur auf Ethernet-Schnittstellen möglich. PROFINET-Schnittstellen werden nicht unterstützt.

5.4.2.5 Systemfunktion _udpDropMulticastGroupMembership()

Übersicht

Diese Systemfunktion dient dazu, eine Multicast-Gruppe an einer ausgewählten Ethernet-Schnittstelle zu verlassen.

```
myRetDINT :=
  _udpDropMulticastGroupMembership(
      multicastIPAddress :=
      ,interfaceIPAddress :=
      ,multicastPort :=
      ,nextCommand :=
);
```

Hinweis

Die UDP Multicast Funktionen (_updAddMulticastGroupMembership(), updDropMulticastGroupMembership()), sind nur auf Ethernet-Schnittstellen möglich. PROFINET-Schnittstellen werden nicht unterstützt.

5.4.3 SIMATIC Kommunikationsbausteine onboard Ethernet-Schnittstelle

5.4.3.1 Übersicht SIMATIC Kommunikationsbausteine

Übersicht

Für die UDP–Kommunikation mit einer SIMATIC und deren onboard-Ethernet-Schnittstellen stehen die gleichen Kommunikationsbausteine wie für die TCP-Kommunikation zur Verfügung. Eine Übersicht finden Sie im Kapitel Übersicht SIMATIC Kommunikationsbausteine (Seite 163).

Der Aufbau und die Parametrierung des UDT65 sollen im Folgenden näher erläutert werden.

Bei der Parametrierung des UDT65 bzw. des sich daraus ableitenden Datenbausteins finden Sie eine ausführliche Erklärung im Kapitel Aufbau und Parametrierung UDT65 (Seite 165). In der Spalte Wert UDP finden Sie die notwendigen Einstellungen für die UDP-Projektierung. Einzig die Fallunterscheidung aktiver und passiver Verbindungsaufbau existiert bei UDP nicht. Der Parameter **active_est** wird deshalb auf FALSE gesetzt.

Siehe auch

Beschreibung der Kommunikationsbausteine (Seite 167)

5.4.4 SIMATIC Kommunikationsbausteine für Ethernet-CP

5.4.4.1 Übersicht S7-Kommunikationsbaustein UDP

Übersicht

Bei den für diesen Anwendungsfall benutzten Funktionen handelt es sich ebenfalls um die bereits beschriebenen S7-Funktionen siehe Übersicht SIMATIC Kommunikationsbaustein (Seite 171)).

Der Projektierungsablauf für eine UDP-Kommunikation ist ähnlich zur TCP-Kommunikation. Einzig bei der Verbindungsprojektierung in NetPro müssen UDP-spezifische Parameter gewählt werden. Außerdem ist es nicht notwendig einen aktiven oder passiven Verbindungsaufbau zu definieren.

Hinweis

Für die **CP 300** beträgt die maximale Sende- und Empfangslänge pro Funktionsaufruf (AG_SEND und AG_RECV) 2048 Bytes.

Für die **CP 400** können ebenfalls die Funktionen AG_SEND und AG_RECV verwendet werden. Allerdings ist hier die übertragbare Datenlänge generell pro Auftrag auf <=240 Bytes beschränkt!

5.4.4.2 Projektierung der Ethernet-CP

Spezifische Projektierungseinstellungen für eine UDP-Kommunikation

Die Projektierung íst ähnlich der Vorgehensweise bei TCP. Im Folgenden sind nur die Unterschiede bei der Projektierung beschrieben. Beachten Sie auch die Vorgehensweise bei TCP (siehe Projektierung der Ethernet-CP (Seite 172)).

1. Im Feld Verbindung wird der gewünschte Verbindungs-Typ **UDP-Verbindung** ausgewählt.

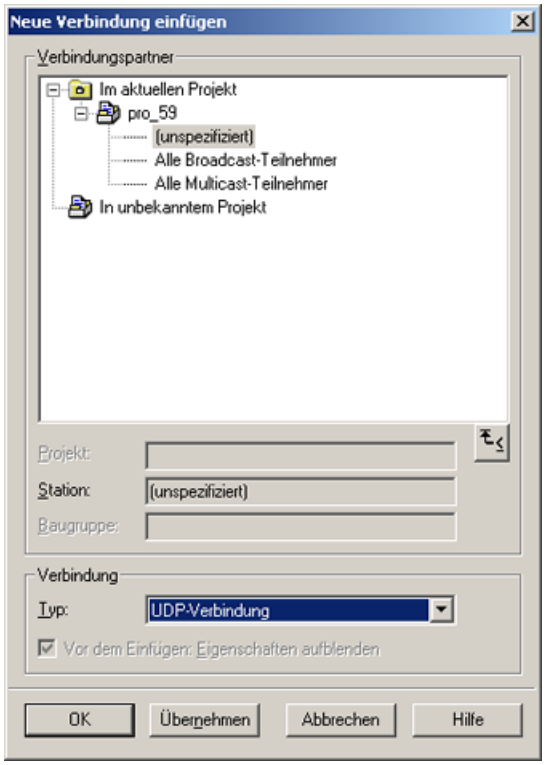


Bild 5-9 Dialog "Neue Verbindung einfügen" mit ausgewählter UDP-Verbindung

2. Im Register **Allgemein** können im Feld **Bausteinparameter** die Parameter ID und LADDR für die S7-Kommunikationsbausteine entnommen werden. Mit ID wird der Kommunikationsverbindung eine eindeutige Referenz zugewiesen. Zudem wird als "**LADDR**" die Adresse des CP angegeben.

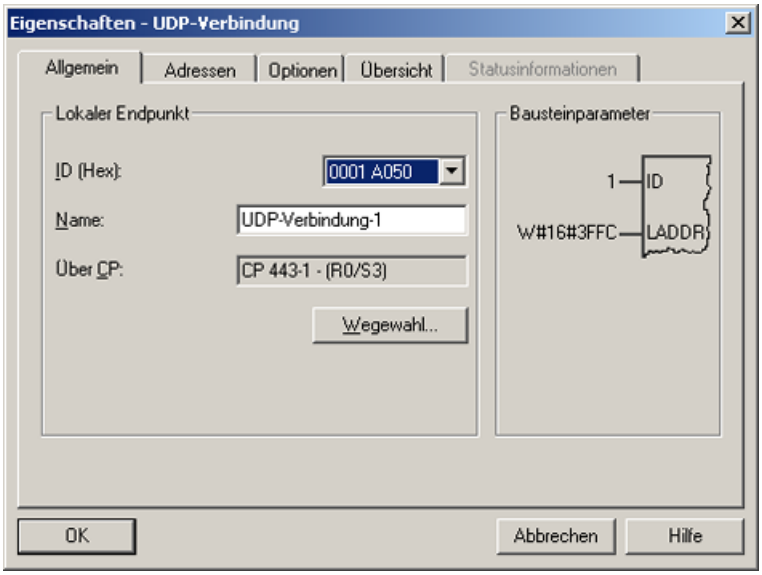


Bild 5-10 Dialog "Eigenschaften - UDP-Verbindung" - Register "Allgemein"

Routing - Kommunikation über Netzwerksgrenzen 6

6.1 Was bedeutet Routing?

Routing bezeichnet das übermitteln von Informationen von einem Netz x in ein Netz y.

Man unterscheidet dabei grundlegend zwischen einem intelligenten, selbstlernenden Routing (z. B. IP-Routing im Internet) und einem Routing gemäß vorher festgelegten Routing-Tabellen (z. B. S7-Routing).

IP-Routing

Beim IP-Routing handelt es sich um ein selbstlernendes Routingverfahren (kann auch manuell durchgeführt werden) ausschließlich in Ethernet Kommunikationsnetzen die mit dem IP-Protokoll arbeiten, wie z. B. das Internet.

Die Funktion wird durch spezielle Router übernommen, die die Informationen anhand der IP-Adresse an benachbarte Netze weiterleiten, wenn die erkannte IP-Adresse im eigenen Netz nicht vorhanden ist.

Hinweis

IP-Routing wird von SIMOTION NICHT unterstützt. Zwischen den Ethernet-Schnittstellen ist ein S7 Routing möglich.

S7-Routing

Beim S7-Routing handelt es sich um ein Routingverfahren, das auf vorher projektierten Routingtabellen aufbaut, dafür aber auch Informationen zwischen unterschiedlichen Kommunikationsnetzen austauschen kann, z. B. zwischen Ethernet, PROFIBUS und MPI. Diese Routingtabellen können als Verschaltungstabellen in NetPro angelegt werden.

Das S7-Routing arbeitet nicht mit der IP-Adresse, sondern mit sogenannten Sub-Netz-IDs innerhalb des S7-Protokolls.

- Informationsübermittlung von Ethernet nach MPI und umgekehrt
- Informationsübermittlung von Ethernet nach PROFIBUS und umgekehrt
- Informationsübermittlung von MPI nach PROFIBUS und umgekehrt
- Informationsübermittlung von Ethernet nach Ethernet (SIMOTION ab V4.1.2, einschließlich PROFINET; CP343, CPU 315-2 PN/DP...)

CIDR Classless Inter-Domain Routing

Classless Inter-Domain Routing (CIDR) beschreibt ein Verfahren zur effizienteren Nutzung des bestehenden 32-Bit-IP-Adress-Raumes (IPv4). Es wurde eingeführt, um die Größe von Routing-Tabellen zu reduzieren und um die verfügbaren Adressbereiche besser auszunutzen. Die Funktion CIDR (classless inter domain routing) beinhaltet Subnetting und Supernetting.

Hinweis

CIDR wird von allen SIMOTION Geräten unterstützt. Bei SIMATIC Geräten beachten Sie bitte die Hinweise in den Gerätehandbüchern.

PG/PC- Zuordnung

Für S7-Routing ist eventuell eine Änderung der PG-Zuordnung nötig. Das können Sie jetzt in der Werkzeugleiste in SIMOTION SCOUT über den Button **PG zuordnen** durchführen. Damit rufen Sie das Eigenschaftsfenster zur PG-Zuordnung auf, indem Sie die Zuordnung anpassen und "aktiv" (S7ONLINE-Zugriff) schalten können.

Hinweis

Weitere Hinweise über Ethernet/PROFINET und die notwendigen Einstellungen für Routing finden Sie in der Produktinformation *SIMOTION SCOUT Übersicht Service- und Diagnosemöglichkeiten*, sowie in der Online-Hilfe zu diesem Thema.

6.2 Projektierung von S7-Routing

Das S7-Routing wird im STEP7 / SIMOTION SCOUT mithilfe der Netzprojektierung "NetPro" projektiert.

Alle in der Netzprojektierung erfassten Stationen können Informationen untereinander austauschen. Dazu müssen Sie in NetPro Verbindungstabellen anlegen. Die erforderlichen Routingtabellen werden beim Übersetzen des Projektes automatisch erzeugt, müssen aber anschließend in alle beteiligten Stationen geladen werden.

6.3 Routing bei SIMOTION

Über Routing können Sie z. B. auf Geräte, die an Subnetze angeschlossen sind, über ein PG/PC ONLINE gehen. Das S7-Routing wird von SIMOTION unterstützt, d. h. Informationen (Engineering-Zugriffe) können durch ein SIMOTION-Gerät von überlagerten Netzwerken wie Ethernet und MPI an unterlagerte Netze wie PROFIBUS oder PROFINET/Ethernet (ab 4.1.2) durchgeroutet werden.

Randbedingungen

Beim Routen von Informationen auf einen taktsynchron betriebenen PROFIBUS sind folgende Randbedingungen in der Betriebsart "DP Slave" zu berücksichtigen:

- Die Funktionen "Äquidistanter Buszyklus" (Voraussetzung für taktsynchrone Anwendungen) und "Aktive Station" (Voraussetzung zum Routen ins unterlagerte Netzsegment) schließen sich gegenseitig aus.
- Der Betrieb aktiver I-Slave am taktsynchronen Bus ist nicht möglich.

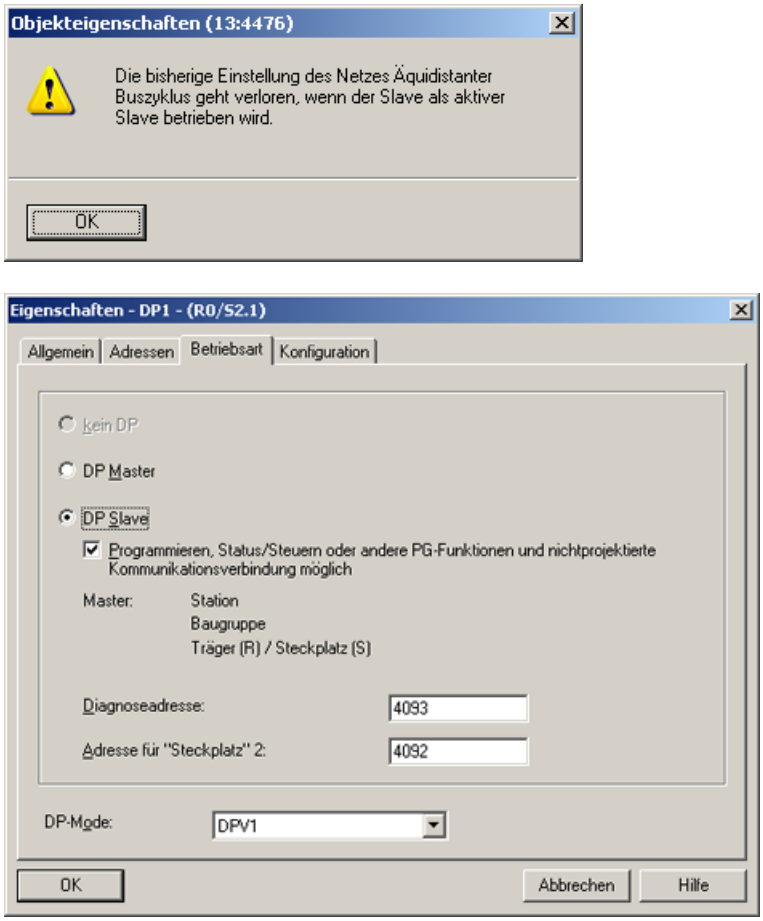


Bild 6-1 Betriebsart DP-Slave: Aktive Station: Test, Inbetriebnahme, Routing

Die Aktivierung des Kontrollkästchens "Programmieren, Status/Steuern oder andere PG-Funktionen …" muss gesetzt sein, wenn Sie z. B. PG-Funktionen, die bei der Inbetriebnahme und beim Testen benötigt werden, häufig über diese Schnittstelle durchführen wollen, oder wenn Sie mit PG-Funktionen (z. B. Starter) auf SINAMICS Antriebe an der kaskadierten, unterlagerten DP-Master Schnittstelle der SIMOTION durchgreifen (S7-Routen) wollen.

Das Aktivieren der Option " Programmieren, Status/Steuern oder andere PG-Funktionen …" bewirkt, dass die Schnittstelle zum aktiven Teilnehmer am PROFIBUS wird (d. h., die Schnittstelle ist am Token-Umlauf des Routing-PROFIBUS beteiligt). Folgende Funktionen sind dann möglich:

- Programmieren (z. B. laden)
- Testen (Status/Steuern)
- S7-Routing (I-Slave als Netzübergang)

Die Busumlaufzeit kann sich verlängern. Bei zeitkritischen Anwendungen, und wenn S7-Routing sowie die Client-Funktionalität für die Kommunikation nicht erforderlich sind, sollte diese Option deshalb nicht aktiviert werden.

Hinweis

Wenn das Kontrollkästchen "Programmieren, Status/Steuern oder andere PG-Funktionen..." nicht aktiviert ist, arbeitet die Schnittstelle nur als Server für zyklische Daten, d. h. S7-Routing ist nicht möglich.

6.4 Routing bei SIMOTION D (Beispiel D4x5 mit CBE30)

Routing zwischen den verschiedenen Schnittstellen

Die beiden Standard Ethernetschnittstellen X120 bzw. X130 der SIMOTION D bilden je ein eigenes Subnetz, alle Ports auf CBE30 bilden ebenfalls ein gemeinsames Subnetz.

- Ein Routing von Subnetz zu Subnetz (IP-Routing) wird nicht unterstützt. Sie können dafür einen externen IP-Router einsetzen.
- Das S7-Routing von einem PROFINET/Ethernet-Subnetz zu einem PROFIBUS ist möglich.

Daraus leiten sich drei Möglichkeiten ab, ein PG/PC oder HMI über S7-Routing an eine SIMOTION D mit CBE30 anzuschließen.

Hinweis

Die Ethernetschnittstellen X120 bzw. X130 und das CBE30 müssen in unterschiedlichen Subnetzen für das S7 Routing liegen.

Engineeringsystem an PROFINET (CBE30)

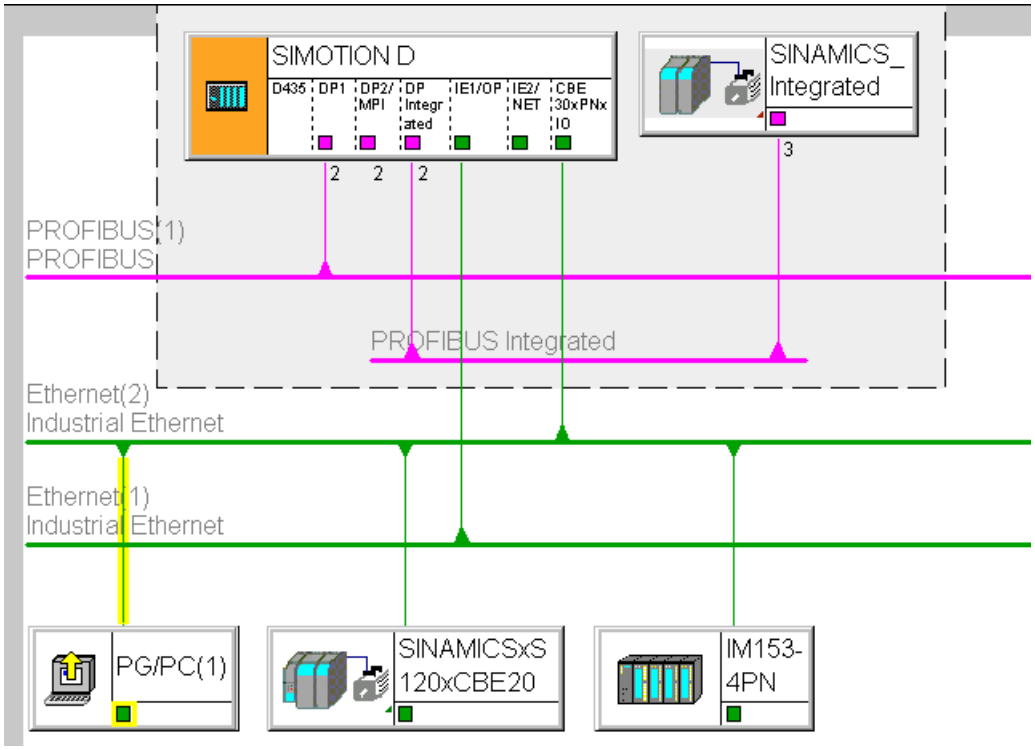


Bild 6-2 Beispiel für PG/PC an CBE30

- S7-Routing auf die (Master-) PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zu den Standard-Ethernetschnittstellen ET1/ET2 (X120, X130) (ab V.4.1.2)
- Zugriff zu den Komponenten am gleichen Subnetz (CBE30) über die Switch-Funktionalität

Engineeringsystem/HMI an PROFIBUS

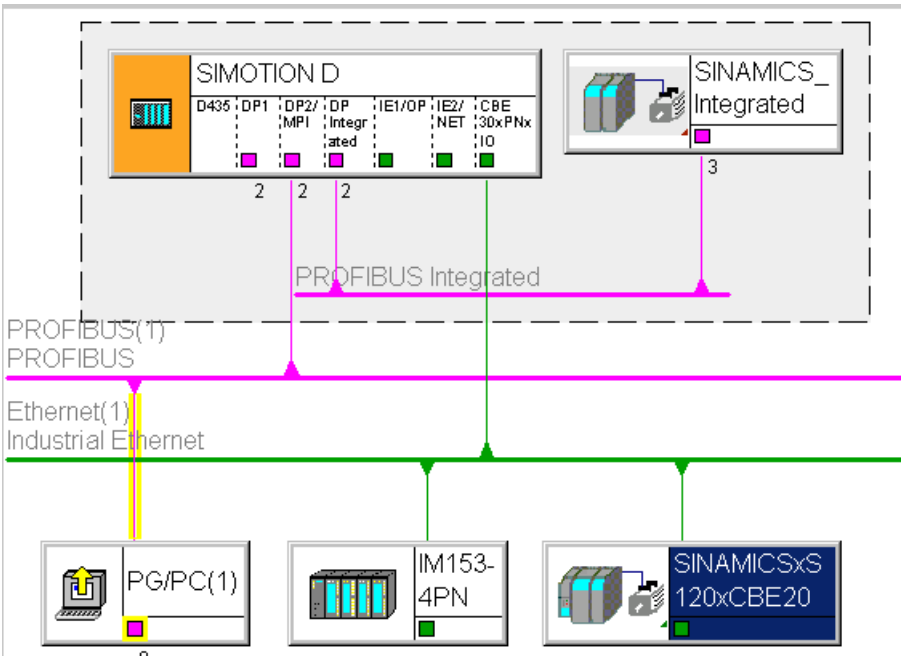


Bild 6-3 Beispiel für PG/PC an PROFIBUS

- S7-Routing zu den anderen (Master-) PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zu X1400 auf dem CBE30
- S7-Routing zu den Standard-Ethernetschnittstellen (X120, X130) (ab V4.1.2)
- Zugriff auf Teilnehmer z. B. HMI am gleichen Netz

Egineeringsystem/HMI an Ethernet

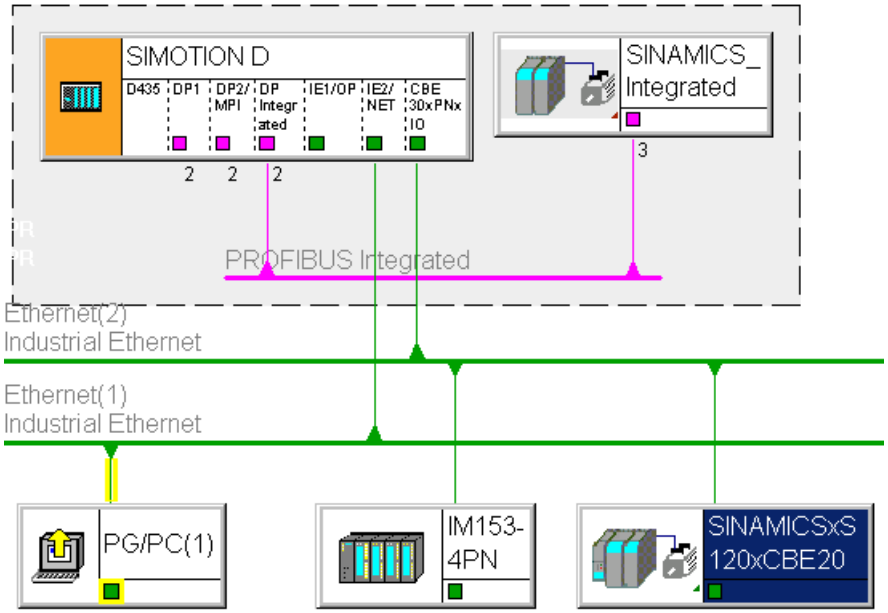


Bild 6-4 Beispiel für PG/PC an Ethernet X120, X130

- S7-Routing zu den anderen (Master-) PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zu X1400 auf dem CBE30
- S7-Routing zwischen den Ethernet-Schnittstellen untereinander
- Zugriff auf Teilnehmer z. B. HMI am gleichen Netz

6.5 Routing bei SIMOTION D4x5-2 (Beispiel D455-2 DP/PN)

Routing zwischen den verschiedenen Schnittstellen

Die beiden Ethernet-Schnittstellen der D4x5-2 DP/PN (X127 P1 bzw. X130 P1) bilden je ein eigenes Subnetz.

Die D4x5-2 DP/PN Onboard PROFINET IO-Schnittstelle (X150, P1-P3) bildet ebenfalls ein eigenes Subnetz. Alle Ports einer PROFINET IO-Schnittstelle gehören immer dem gleichen Subnetz an.

- Ein Routing von Subnetz zu Subnetz (IP-Routing) wird nicht unterstützt. Sie können dafür einen externen IP-Router einsetzen.
- Das S7-Routing von einem PROFINET/Ethernet-Subnetz zu PROFIBUS ist möglich.

Daraus leiten sich folgende Möglichkeiten ab, ein PG/PC oder HMI-Gerät über S7-Routing an eine SIMOTION D anzuschließen.

Engineeringsystem/HMI an PROFINET

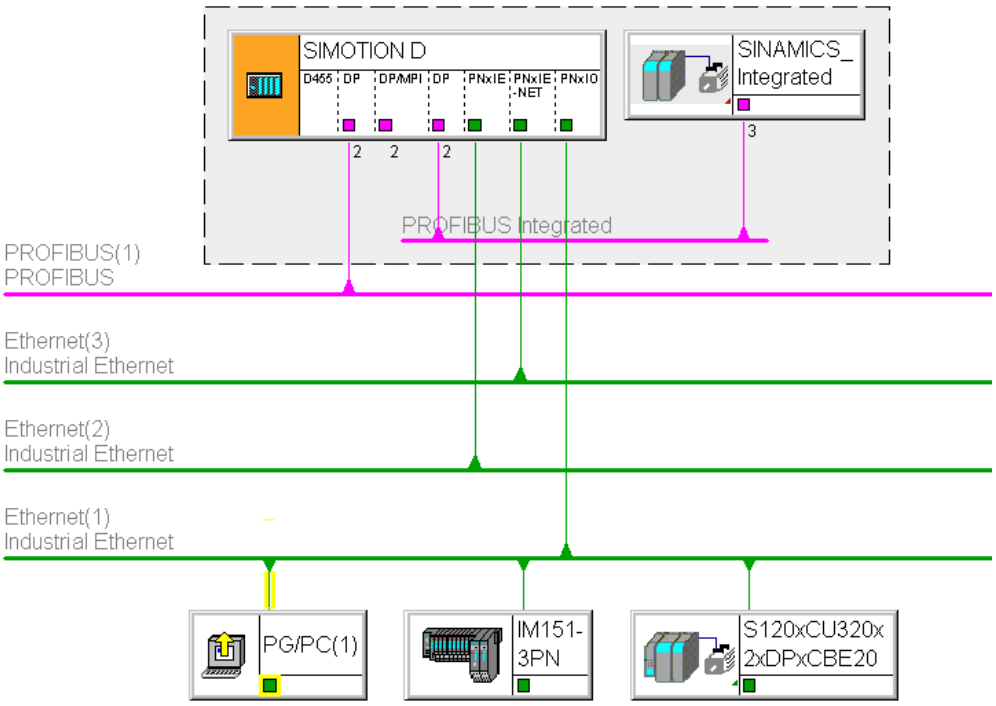


Bild 6-5 Beispiel für PG/PC an PROFINET-Schnittstelle (PNxIO, X150)

- S7-Routing auf die (Master-) PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zu den Ethernetschnittstellen PN/IE (X127 P1) und PN/IE-NET (X130 P1)
- Zugriff zu den Komponenten am gleichen Subnetz über die Switch-Funktionalität der PROFINET IO-Schnittstelle

Engineeringsystem/HMI an PROFIBUS

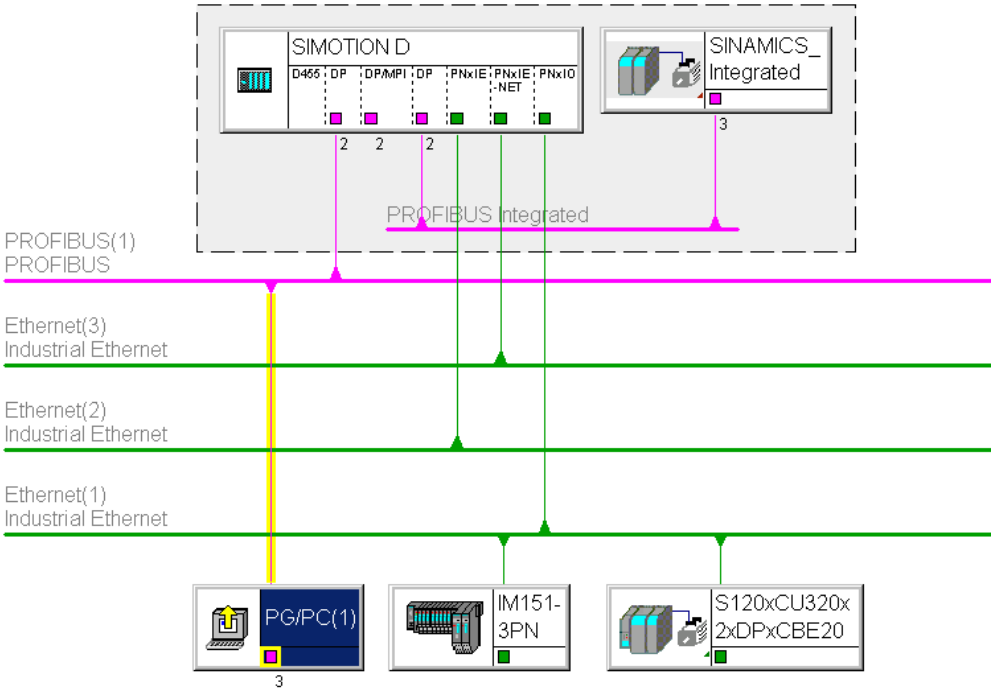


Bild 6-6 Beispiel für PG/PC an PROFIBUS-Schnittstelle (DP, X126)

- S7-Routing zu den anderen (Master-) PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zur Onboard PROFINET IO-Schnittstelle (X150, P1-P3)
- S7-Routing zu den Ethernetschnittstellen PN/IE (X127 P1) und PN/IE-NET (X130 P1)

Engineeringsystem/HMI an Ethernet

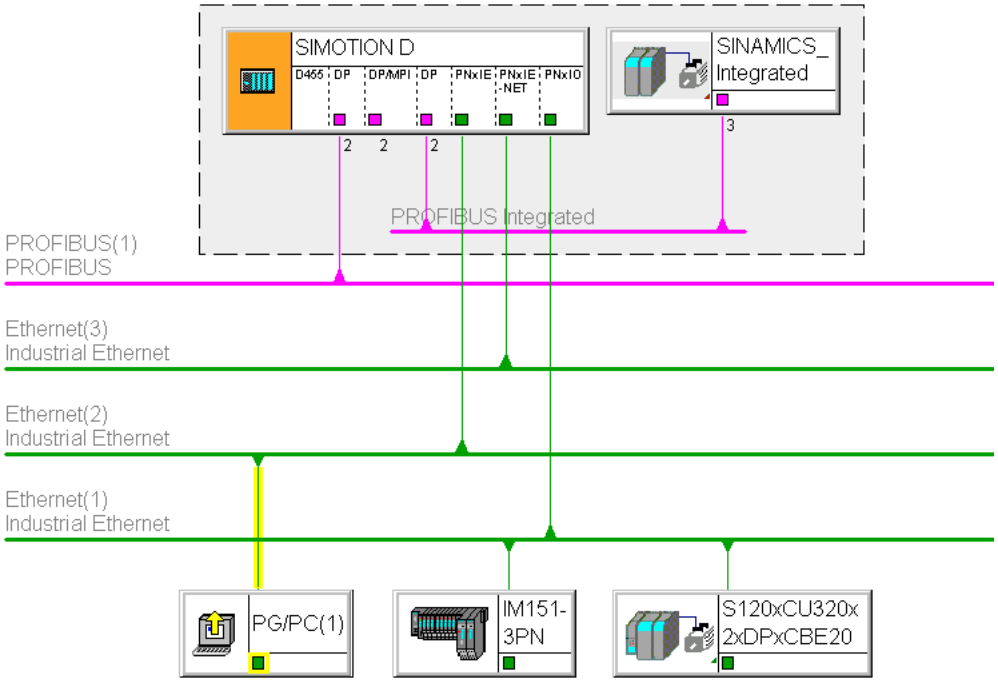


Bild 6-7 Beispiel für PG/PC an Ethernet-Schnittstelle (PNxIE, X127)

- S7-Routing zu den anderen (Master-)PROFIBUS-Schnittstellen (nur wenn projektiert)
- S7-Routing zum PROFIBUS Integrated
- S7-Routing zur Onboard PROFINET IO-Schnittstelle (X150, P1-P3)
- S7-Routing zwischen den Ethernet-Schnittstellen untereinander

6.6 Routing bei SIMOTION D zum SINAMICS integrated

S7-Routing zum internen PROFIBUS auf SINAMICS Integrated

Alle SIMOTION D haben eine SINAMICS-Antriebsregelung integriert. Um auf Antriebs-Parameter zugreifen zu können, müssen die Telegramme von den externen SIMOTION D-Schnittstellen auf den internen PROFIBUS DP durchgeroutet werden. Über S7-Routing können Sie auf den integrierten PROFIBUS zugreifen. Der interne PROFIBUS DP bildet dabei ein eigenes Subnetz. Dies ist insbesondere bei der Kommunikation auf mehrere Routing-Knoten zu beachten.

6.7 Routing bei SIMOTION P350

Beschreibung

S7-Routing ist möglich:

- von PROFIBUS (IsoPROFIBUS-Board) auf PROFINET-Subnetz an MCI-PN-BOARD
- von PROFINET-Subnetz an MCI-PN-BOARD auf PROFIBUS (IsoPROFIBUS-Board)
- von Scout auf SIMOTION P über Softbus durch Run-Time hindurch auf PN-Geräte am MCI-PN-Board und IsoPROFIBUS-Board
- von Onboard-Ethernet-Schnittstellen auf PROFIBUS (IsoPROFIBUS-Board) und auf PROFINET
- S7-Routing zwischen den Ethernet-Schnittstellen untereinander

IP-Routing ist über die Ethernet-Schnittstellen der P350 **nicht** möglich.

Routing von PROFIBUS auf PROFINET

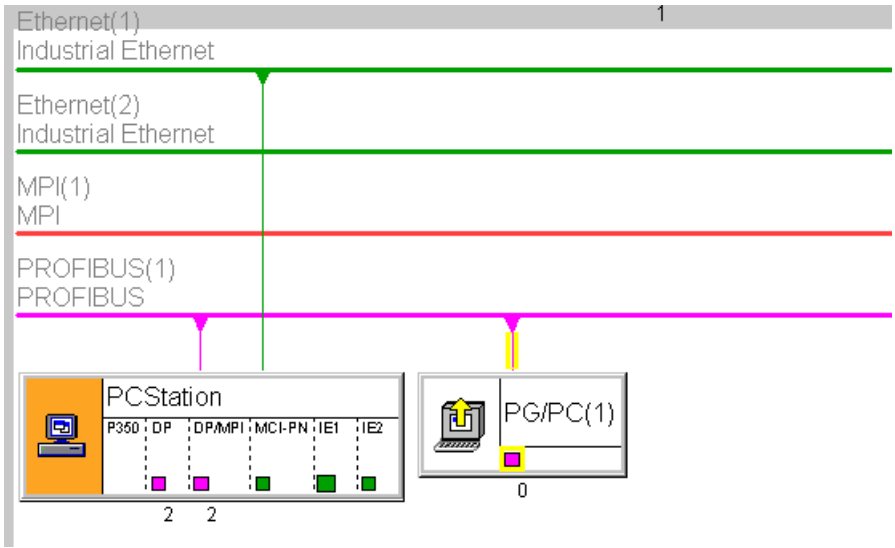


Bild 6-8 Beispiel für P350 Routing von PROFIBUS nach PROFINET

Routing von PROFINET auf PROFIBUS

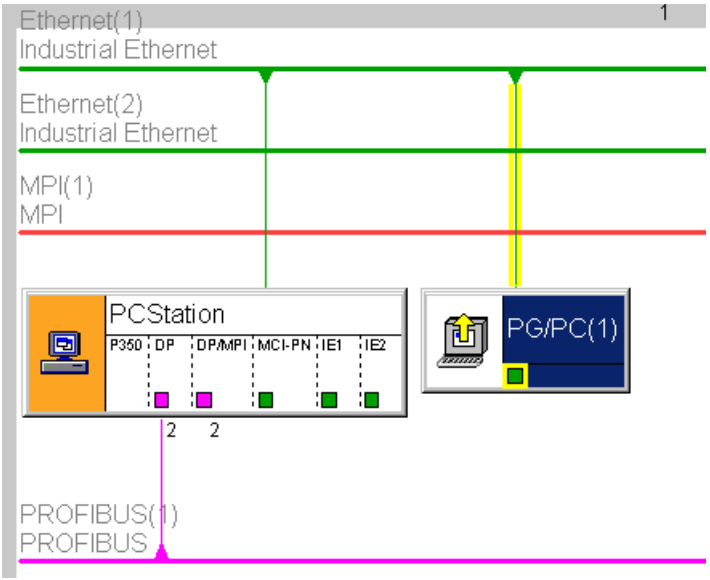


Bild 6-9 Beispiel für P350 Routing von PROFINET auf PROFIBUS

6.8 Routing bei SIMOTION P320

Beschreibung

S7-Routing ist möglich:

- von der Onboard-Ethernet-Schnittstelle zum PROFINET-Subnetz und den Antriebsgeräten bzw. SIMOTION Geräten am PROFINET-Subnetz

Routing von Ethernet auf PROFINET

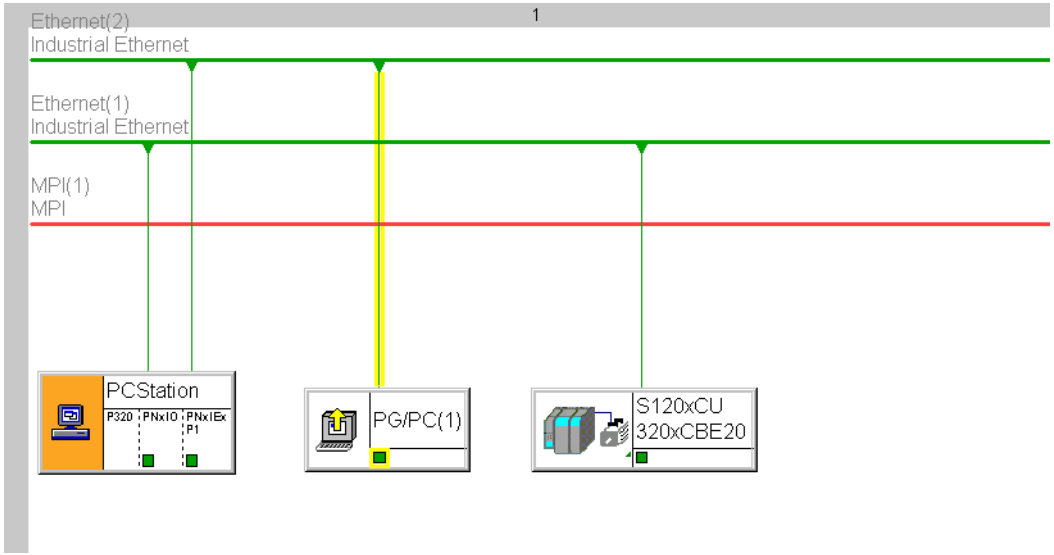


Bild 6-10 Routing bei SIMOTION P320

SIMOTION IT 7

7.1 SIMOTION IT - Übersicht

Beschreibung

SIMOTION IT bietet die Möglichkeit über Standard-Webdienste (HTTP) auf die SIMOTION Steuerung zuzugreifen.

Dabei bieten sich folgende Vorteile.

- Ortsunabhängige offene Diagnose / Prozessbeobachtung
- Client-Gerät unabhängig vom Betriebssystem (Windows, Linux..)
- Standardisierte Kommunikations-Schnittstelle für herstellerspezifische Tools
- Unabhängig vom Engineeringsystem
- Kein Versionskonflikt zwischen Client-Applikation und SIMOTION RT (Runtime)
- Serieninbetriebnahme ohne Engineeringsystem

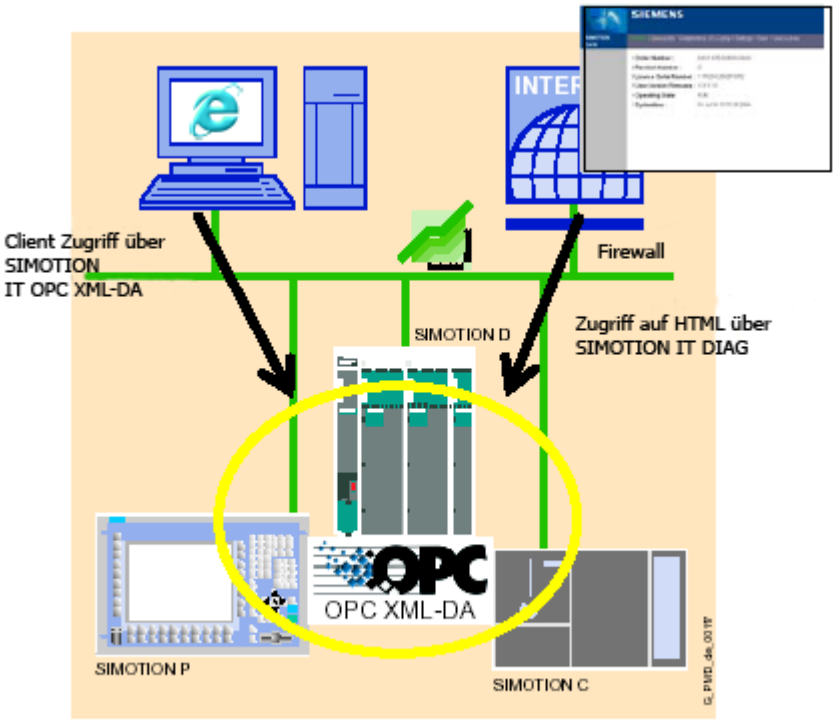


Bild 7-1 SIMOTION IT Übersicht

SIMOTION IT besteht aus folgenden Funktionspaketen:

- SIMOTION IT DIAG
- SIMOTION IT OPC XML - DA
- Trace via SOAP
- Dateidownload über FTP (File Transfer Protokoll)
- SIMOTION IT Virtual Machine

Weiterführende Literatur

Eine detaillierte Beschreibung der SIMOTION IT Produkte finden Sie in der Produktinformation *SIMOTION IT Ethernet basierende HMI- und Diagnosefunktionen* auf der DVD SIMOTION SCOUT Dokumentation.

Weitereführende Informationen zum SIMOTION VM finden Sie im *Programmierhandbuch SIMOTION – IT Virtual Machine*.

Siehe auch

Webzugriff auf SIMOTION (Seite 203)

SIMOTION IT DIAG (Seite 204)

SIMOTION IT OPC XML-DA (Seite 206)

7.2 Webzugriff auf SIMOTION

Beschreibung

Die nachfolgende Grafik zeigt die verschiedenen Möglichkeiten auf die Daten in einer SIMOTION-Baugruppe zuzugreifen.

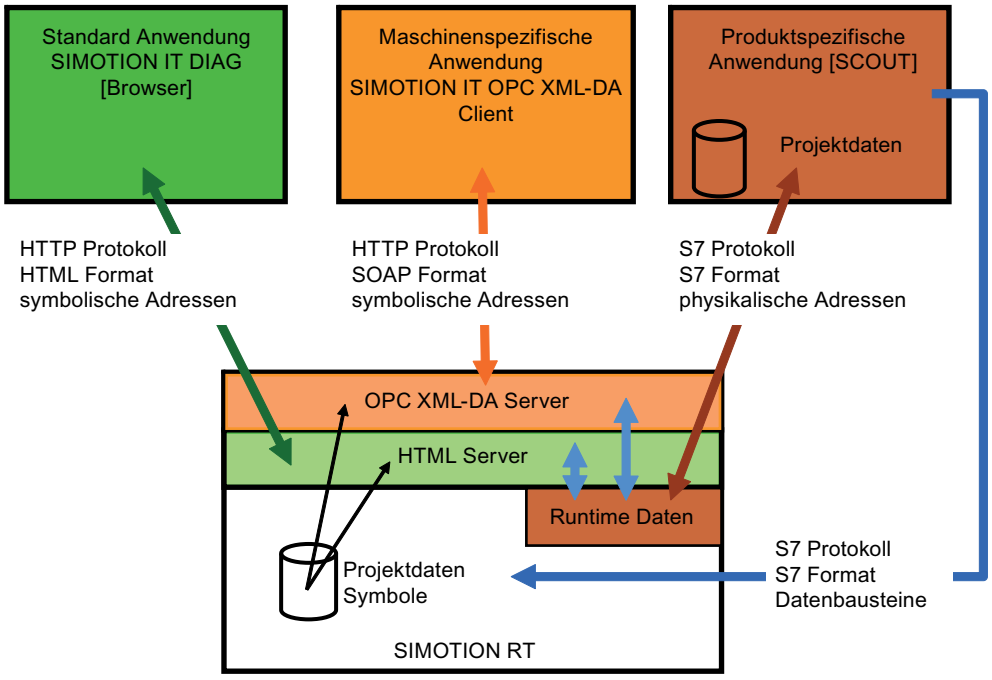


Bild 7-2 Zugriff auf SIMOTION

Siehe auch

SIMOTION IT DIAG (Seite 204)

SIMOTION IT OPC XML-DA (Seite 206)

7.3 SIMOTION IT DIAG

Beschreibung

SIMOTION IT DIAG bietet die Möglichkeit, von einem PC aus mit einem beliebigen Internetbrowser auf die HTML-Seiten in SIMOTION zuzugreifen.

Standarddiagnoseseiten

SIMOTION bietet folgende Standarddiagnoseseiten:

- Startseite
- Device Info/IP-Config
  (Informationen über Firmware, Geräte, Gerätekomponenten, Technologiepakete und Daten der der Ethernet-Schnittstelle des SIMOTION-Gerätes)
- Diagnostics
  (CPU-Auslastung, Speicherverbrauch, Betriebszustand, Tasklaufzeitenanzeige, Trace für Geräte und System, Serviceoverview)
- Message&Logs
  (Diagnosepuffer, Alarme SIMOTION und Drives, Sys- und Userlog)
- Machine Overview
  (Module undTopologie einer Maschine sowie Informationen über die Hardwarekonfiguration)
- Manage Config
  (IT DIAG Konfigurationen laden, Deviceupdates, Daten vom Gerät speichern)
- Settings
  (Einstellen der Zeitzone, Betriebszustände schalten, Anzeige benutzerdefinierten Seiten verändern)
- Files
  (Zugriff auf das SIMOTION-Filesystem, Up- und Download von Dateien, Erzeugen von Ordnern und Ablegen von weiteren Daten, z. B. Dokumentation)

Vereinfachte Standardseiten

Für die optimierte Darstellung von IT DIAG Seiten auf Geräten, wie Handy oder PDA, wird ab der Version 4.1.3 ein Satz von speziellen Seiten bereitgestellt. Diese enthalten Informationen der Standardseiten in vereinfachter Darstellung. Die Startseite der vereinfachten Standardseiten ist unter der Adresse http://<IPAddr>/BASIC erreichbar.

Konfiguration über WebCfg.xml

Es gibt zwei Dateien mit denen IT DIAG konfiguriert werden kann:

- WebCfg.xml
- WebCfgFrame.xml

Über die Konfigurationsdatei WebCfg.xml werden anwenderrelevante Einstellungen im Webserver konfiguriert. Die WebCfgFrame.xml enthält die IT DIAG Einstellungen des Herstellers.

Anwenderdefinierte Seiten

SIMOTION IT DIAG bietet die Möglichkeit individuell gestaltete Webseiten zu integrieren. Für den Zugriff auf SIMOTION Variablen stehen zwei Mechanismen zur Verfügung:

- die Javascript Bibliotheken opcxml.js und appl.js
- die MiniWeb Server Language (MWSL)

Anwenderdefinierten Seiten können im Bereich "User´s Area" dargestellt werden. Hierfür müssen diese im Ordner FILES des SIMOTION Dateisystems abgelegt sein.

Trace via SOAP

Der WebTrace ist identisch mit dem SIMOTION SCOUT Trace. Der einzige Unterschied besteht nur in der Ausgabe des Datenformats.

Das Funktionspaket "Trace via SOAP" ermöglicht es, SIMOTION Variablen in einen Puffer zu schreiben. Die Werte werden in eine Datei gepackt und können per HTTP Request asynchron abgeholt werden. Die Nutzung dieser Schnittstelle erfordert grundsätzlich eine Applikation als Client.

Der Trace kann über ein Webinterface parametriert und die Aufzeichnung über einen TraceViewer sofort betrachtet werden.

Variablenzugriff

Der Variablenzugriff ist für die SIMOTION IT Anwendungen über Variablenprovider realisiert. Zurzeit existieren vier Variablen Provider.

- MinWeb
- SIMOTION
- SIMOTION diagnostics
- UserConfig

Dieser erlaubt es, auf folgende Variablen zuzugreifen:

- Device-Systemvariablen
- TO-Systemvariablen
- Globale Unit-Variablen aus dem Interfaceabschnitt
- TO-Konfigurationsdaten
- ab 4.2 auch geräteglobale Variablen und I/O-Variablen
- Antriebsparameter
- Einstellen des Betriebszustands, Ausführen von RamToRom, Ausführen von ActiveToRom

- Technologische Alarme
- Diagnosepuffer

Sichere HTTPS Verbindung

Durch das Secure Socket Layer Protokol (SSL) wird eine verschlüsselte Datenübertragung zwischen einem Client und dem SIMOTION Gerät ermöglicht. Das Secure Socket Layer Protokoll bildet die Basis für HTTPS Zugriffe. Der verschlüsselte Zugriff kann sowohl über SIMOTION IT OPC XML-DA als auch über SIMOTION IT DIAG erfolgen.

7.4 SIMOTION IT OPC XML-DA

Beschreibung

Der SIMOTION IT OPC XML-DA Server ermöglicht via Ethernet auf Daten und Betriebszustände des SIMOTION Geräts zuzugreifen. Für SIMOTION IT OPC XML-DA ist ab V4.2 keine eigene Lizenz mehr erforderlich.

OPC ist die Abkürzung für open connectivity und bezeichnet eine Standardschnittstelle für die Kommunikation in der Automatisierungstechnik. Mit OPC XML-DA ist es möglich, über Ethernet basierende Standardtelegramme mit einer Steuerung zu kommunizieren. Zur Befehlsübermittlung dient das Kommunikationsprotokoll SOAP (simple object access protocol).

Eine auf einem Client PC erstellte kundenspezifische Applikation, die z. B. mit der Programmiersprache C#, Visual Basic oder Java programmiert ist, nutzt die Dienste und Eigenschaften von SIMOTION IT OPC XML-DA:

- Offene Kommunikation über HTTP, SOAP, OPC-XML zwischen Client und SIMOTION Gerät.
- Setzt auf die Spezifikation OPC XML-DA 1.0 der OPC-Foundation auf
- Zugriff auf SIMOTION Variablen
  - Lesen/Schreiben
  - Zyklische Lesen Subscriptions
  - Browsen
- Trace via SOAP; diese Funktion ist eine Erweiterung der OPC-Spezifikation
- Clients auf beliebiger HW mit unterschiedlichen Betriebssystemen (Windows, Linux, ..)
- Erstellung von Client-Applikationen mit C#, Java, C++. Die Applikation, über die Sie auf den SIMOTION OPC XML-DA Server zugreifen möchten, müssen Sie selbst implementieren.
- Zugriffschutz über Benutzergruppen, Benutzerkennung und Passwort

Die folgende Grafik zeigt schematisch den Zugriff auf den OPC XML-DA Server.

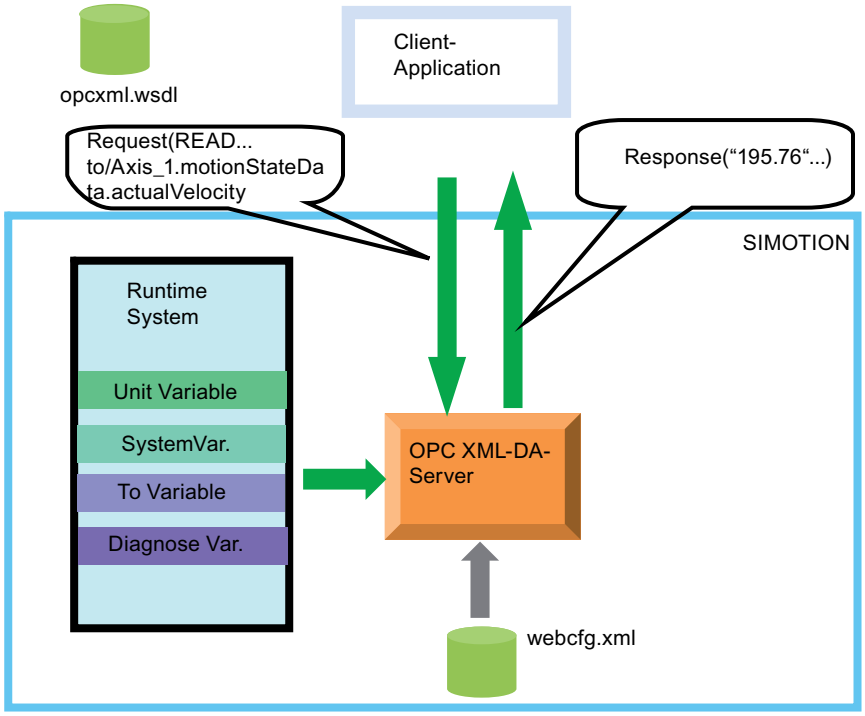


Bild 7-3 Zugriff auf den OPC XML-DA Server

7.5 FTP Datentransfer

Filezugriff über FTP

Mit Hilfe des FTP-Servers der SIMOTION Steuerung können Sie gezielt auf das SIMOTION Dateisystem zugreifen. FTP ist über einen Zugriffschutz verriegelt.

Über FTP können Sie z. B. Firmware Updates durchführen oder anwenderdefinierte Seiten laden.

Der FTP-Dienst erfordert keine eigene Lizenz.

PROFIsafe

8

8.1 Kommunikationsbeziehungen bei Drive Based Safety

Beschreibung

Die Drive Based Safety Funktionen im Antrieb können entweder über sichere Klemmen direkt am Antrieb oder über PROFIBUS/PROFINET von einer fehlersicheren Steuerung (F-Steuerung) aus gesteuert werden.

Die Ansteuersignale zu den Drive Based Safety Funktionen und die Rückmeldesignale zum Safety Funktionsstatus sind sicherheitsrelevant und müssen über einen Kommunikationskanal übertragen werden, der mit dem PROFIsafe Protokoll gesichert ist. Das allgemeine Szenario der Wechselwirkung zwischen den verschiedenen Steuerungs- und Antriebsprozessen sowie die hierfür notwendigen Kommunikationsbeziehungen zwischen diesen zeigt schematisch das folgende Bild.

Signalfluss für die An- und Abwahl der Drive Based Safety Funktionen
und deren Signalisierung an den Antriebsteuerungsprozess

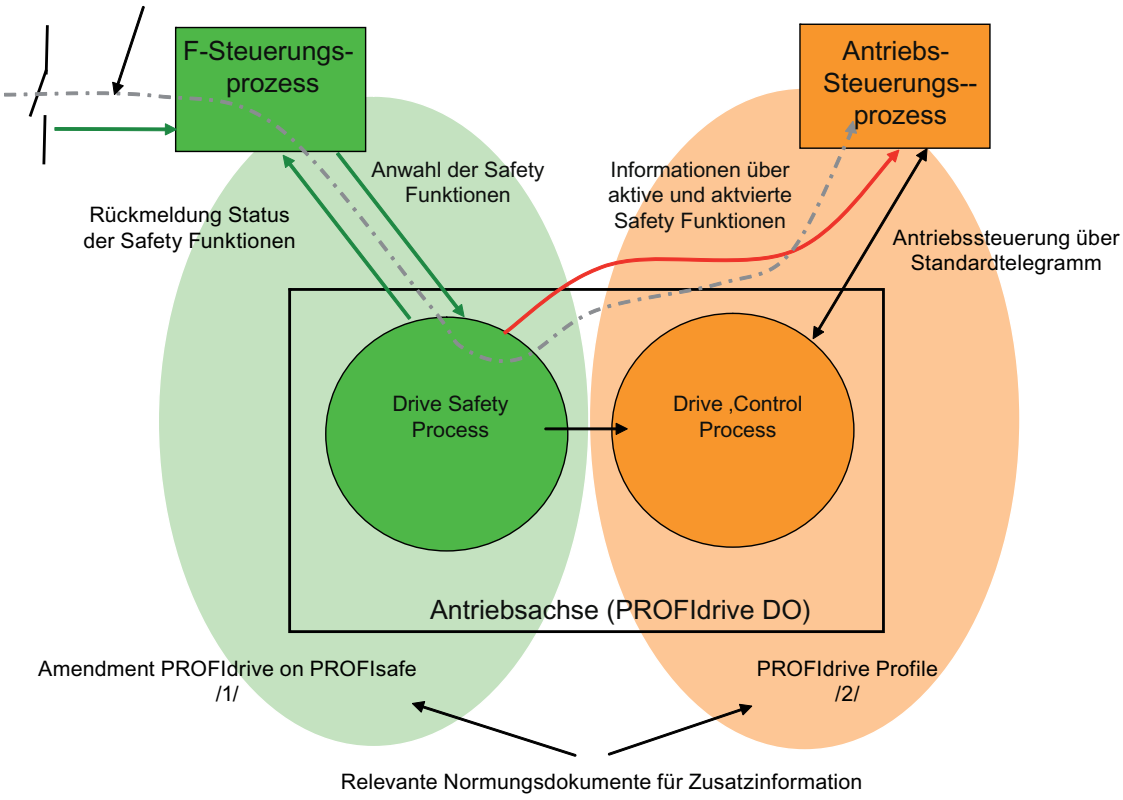


Bild 8-1 Kommunikationsbeziehungen bei Drive Based Safety

Die Schnittstellen des "Drive Safety Prozess" zur F-Steuerung und zur Antriebs-Steuerung ist eine PROFIdrive Schnittstelle und deren Funktionalität in /1/ und /2/ definiert.

Die Einleitung und Überwachung einer Drive Based Safety Funktion erfolgt durch die F-Steuerung über den mit PROFIsafe gesicherten Übertragungskanal von der F-Steuerung zum Antrieb (Antriebsachse). Die jeweiligen Zustände der Drive Based Safety Funktionen im Antrieb haben auch Rückwirkungen auf die Antriebsschnittstelle zur Antriebssteuerung, da bei einigen Drive Based Safety Funktionen die Antriebshoheit zwischen Antriebssteuerung und Drive Safety Process wechselt.

Zur effektiven Koordination von F-Steuerung und Antriebssteuerung ist deshalb zusätzlich ein Informationskanal vom "Drive Safety Process" zur Antriebssteuerung notwendig, damit die Antriebssteuerung auf die gewünschten bzw. aktivierten Drive Based Safety Funktionen entsprechend reagieren kann.

8.2 Telegramme und Signale bei Drive Based Safety

Beschreibung

Da SIMOTION über keine sichere Logikfunktion verfügt, kann der F-Steuerungsprozess nicht von SIMOTION wahrgenommen werden, sondern wird mithilfe eines zweiten Controllers mit F-Funktionalität, typischerweise eine SIMATIC F-CPU, realisiert. Folgendes Bild zeigt diese Konstellation am Beispiel einer SINAMICS-Achse. Vom Antriebs-DO geht ein mit PROFIsafe gesicherter Kommunikationskanal zur SIMATIC F-CPU. Für diesen Kommunikationskanal steht in der Norm das Standardtelegramm 30 zur Verfügung, welches u. a. aus dem Safety Steuer- und Statuswort besteht. Durch das Anwenderprogramm auf der F-CPU werden über das Safety-Steuerwort die projektierten Drive Based Safety Funktionen im Antrieb (Drive safety process) an- oder abgewählt. Die Rückmeldung der aktiven Safety Funktionen erfolgt in den Input-Daten über das Safety-Zustandswort an die F-CPU. Die Rückmeldung der aktiven Safety Funktionen erfolgt von einem sicheren Prozess im Antrieb über einen gesicherten Kommunikationskanal und kann damit zur Freischaltung von Schutzräumen und Türen durch die F-CPU verwendet werden.

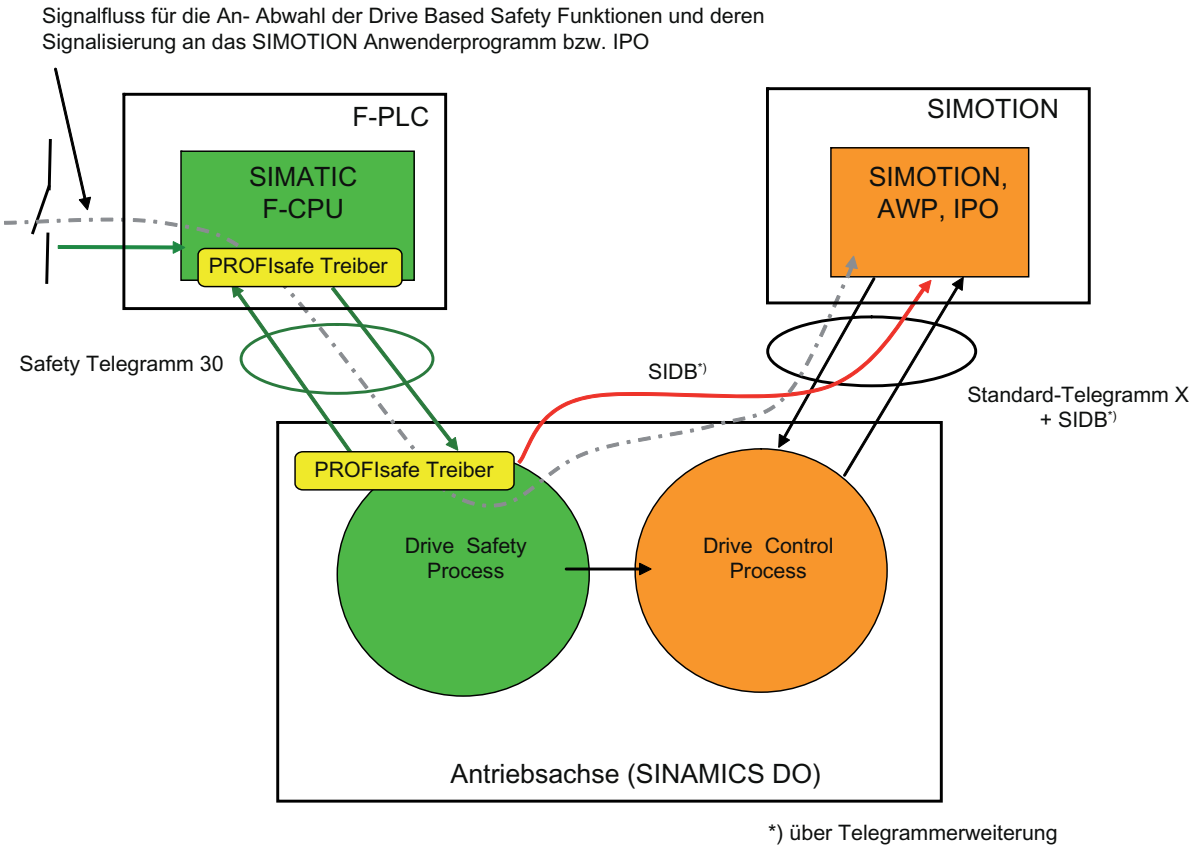


Bild 8-2 Telegramme und Signale bei Drive Based Safety

Parallel zur Steuerung der Safety Funktionen im Antrieb über den PROFIsafe Kanal, gibt es noch einen optionalen Safety Informationskanal (SIDB Safety Data Block), über den Aktivierungs- und Statuszustände der Drive Based Safety Funktionen vom Antrieb an SIMOTION übertragen werden. Dieser Informationskanal ist ungesichert und wird in der Praxis durch eine Telegrammverlängerung des Standardtelegramms um den SIDB-Datenblock realisiert. Zweck des Safety Informationskanals ist die optionale Einbindung der Antriebsbewegungssteuerung (IPO, SERVO) sowie des gesamten Anwenderprogramms (AWP) in den höherprioren Ablauf der Drive Based Safety Funktionen sowie des Anwenderprogramms der F-Steuerung. Typische SIMOTION Reaktionen sind z. B.:

- Erkennen einer safetybedingten autonomen Handlung des Antriebes (z. B. Bremsrampe bei SS1 und SS2 und wechseln in den Nachführbetrieb).
- Erkennen einer Safety Funktionsanwahl und daraus abgeleiteter programmierter SIMOTION-Reaktion auf die angewählte Safety Funktion (z. B. steuerungsbasierte Bremsrampe bzw. Geschwindigkeitsabsenkung bei SOS und SLS).

8.3 F-Proxy Funktionen von SIMOTION

Beschreibung

Zur PROFIsafe-Anbindung der integrierten Antriebe von SIMOTION D, sowie von SINAMICS Antrieben, die von SIMOTION gesteuert werden, aber z. B. in einer anderen Kommunikationsdomain als die F-CPU liegen, verfügt SIMOTION über eine integrierte F-Proxy Funktionalität. Der F-Proxy ermöglicht ein transparentes Durchleiten der Safety-Telegramme vom SIMOTION I-Slave bzw. I-Device Interface zum jeweiligen SIMOTION-Master- bzw. Controller-Interface, an dem der Antrieb projektiert ist. Trotz der Durchleitefunktion von SIMOTION ist die PROFIsafe-Kommunikation zwischen F-CPU und Antrieb gesichert, da die PROFIsafe-Treiber in den Endpunkten (F-CPU, Antrieb) die Kommunikation sicher überwachen.

Zur Nutzung der F-Proxy-Funktionalität sind die beiden Kommunikationsstrecken, von der F-CPU zu SIMOTION und von SIMOTION zum Antrieb getrennt zu projektieren.

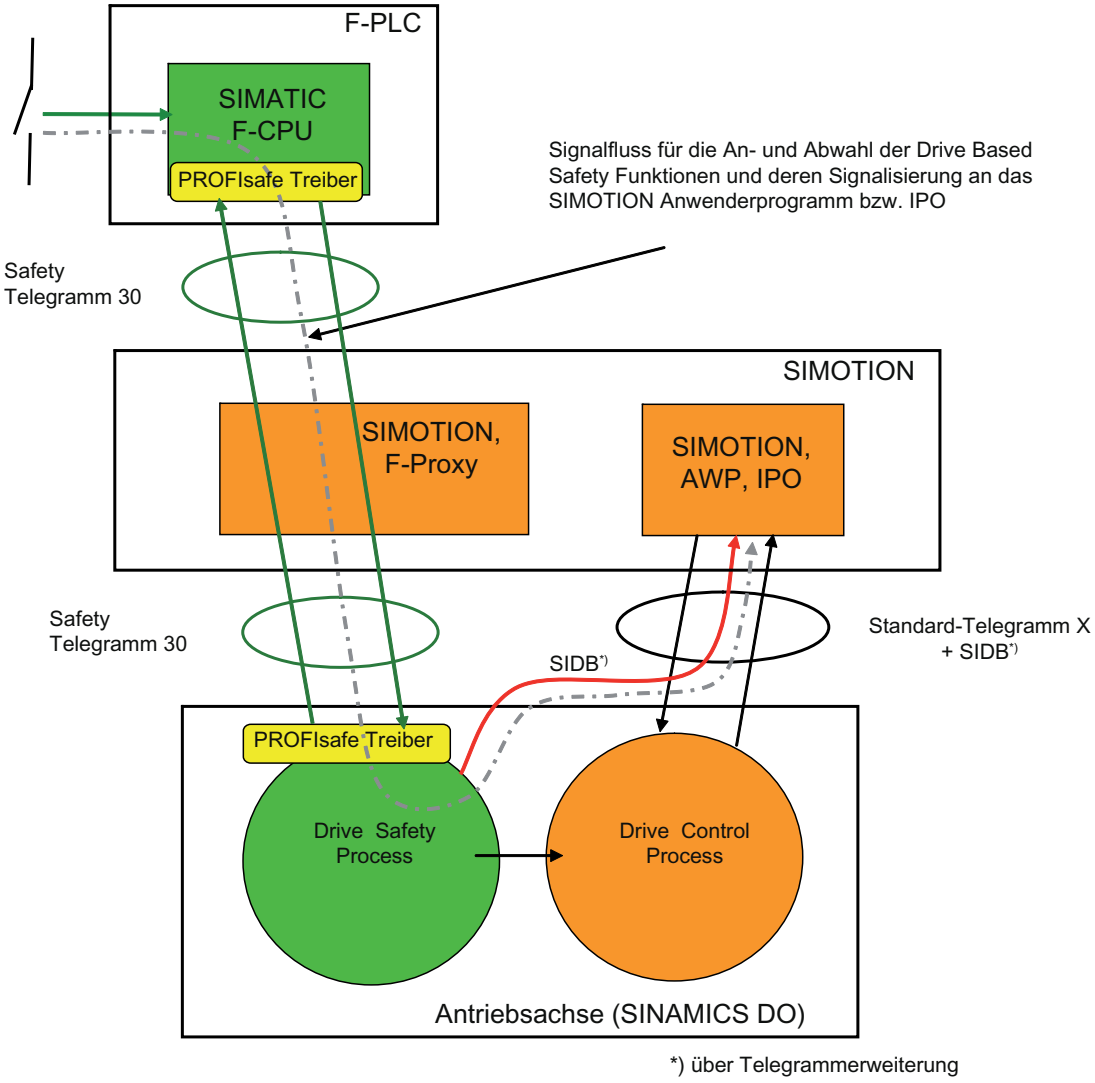


Bild 8-3 Durchleitung des PROFIsafe Kanals mit dem F-Proxy

8.4 Eigenschaften PROFIsafe bei der Projektierung

Mechanismen bei PROFIsafe

Grundsätzlich kann die PROFIsafe Kommunikation sowohl über PROFINET wie auch über PROFIBUS erfolgen. Dazu stehen die Mechanismen des F-Proxys bzw. eine Querverkehr-Variante zur Verfügung.

Gleich bei allen Mechanismen ist, dass die F-Daten von der F-CPU verwaltet werden und die Motion-Control Daten von der SIMOTION CPU. Die notwendigen Eigenschaften des PROFIsafe werden in HW Konfig bei den F-Parametern eingestellt.

Hinweis

Für SIMOTION Projekte (RT-Version) < V4.2 ist die gesicherte PROFIsafe Kommunikation nur über PROFIBUS möglich. Bei SIMOTION Geräten < V4.2 mit PROFIBUS und PROFINET Schnittstellen können Sie PROFIsafe daher nur über die PROFIBUS-Schnittstelle realisieren und die Motion-Control Aufgaben z. B. über die PROFINET-Schnittstelle. Der Mischbetrieb (Bewegungsführung über PROFINET und PROFIsafe über PROFIBUS) ist jedoch nur bei den integrierten SIMOTION D Antrieben möglich, bei Verwendung von externen Antrieben (z. B. S120 CU320-2) an einer SIMOTION CPU ist dies nicht zulässig.

Übersicht F-Parameter PROFIsafe bei der Projektierung

Den Dialog PROFIsafe erhalten Sie, wenn Sie in HW Konfig im Baugruppenträger des Antriebs den Eintrag **PROFIsafe** doppelklicken.

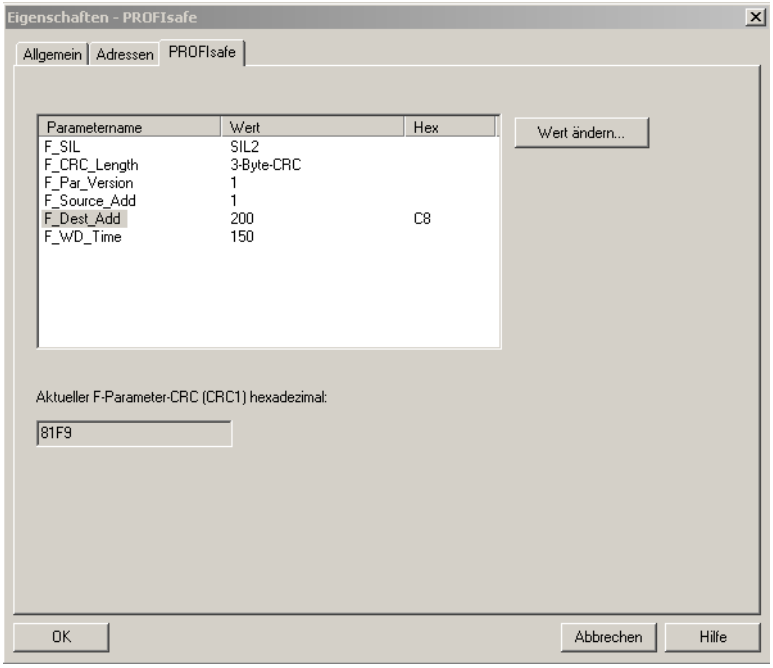


Bild 8-4 Eigenschaften PROFIsafe am Beispiel I-Device-F-Proxy

F_CRC_Length und F_Par_Version

Kennzeichnet neben der Länge der F-Nutzdaten auch den PROFIsafe MODE.

- PROFIsafe V1-MODE

  F_CRC_Length = 2 bis Nutzdatenlänge 12 Bytes

  F_CRC_Length = 3 ab Nutzdatenlänge 13 Bytes

  F_Par_Version = 0

- PROFIsafe V2-MODE

  F_CRC_Length = 3 bis Nutzdatenlänge 12 Bytes

  F_CRC_Length = 4 ab Nutzdatenlänge 13 Bytes

  F_Par_Version = 1

Hinweis

PROFIsafe V2-Mode wird nicht von allen Geräten bzw. Firmwareversionen unterstützt. Vergewissern Sie sich vor der Projektierung, welcher Mode ab welcher Version möglich ist.

SINAMICS G120 CU240D DP F und SINAMICS G120 CU240S DP F unterstützen V2-Mode erst ab der Firmwareversion V3.2.

F_Dest_Add: 1-65534
F_Dest_Add legt die PROFIsafe-Zieladresse des Antriebsobjektes fest.
Der Wert kann beliebig innerhalb des Bereichs liegen, muss aber in der Safety-Projektierung des Antriebs im SINAMICS-Antriebsgerät zusätzlich manuell bzw. bei der Konfiguration von Safety Integrated eingetragen werden (Antriebsparameter p9610/p9810).

F_WD_Time: 10- 65535
Innerhalb der Überwachungszeit muss ein gültiges aktuelles Sicherheitstelegramm von der F-CPU ankommen. Andernfalls geht der Antrieb in einen sicheren Zustand.
Die Überwachungszeit sollte so hoch gewählt werden, dass Telegrammverzögerungen durch die Kommunikation toleriert werden, aber im Fehlerfall (z. B. Unterbrechung der Kommunikationsverbindung) die Fehlerreaktion möglichst schnell ausgeführt wird.

Weitere Informationen zu den F-Parametern finden Sie in der Online-Hilfe des Dialogs **PROFIsafe.**

8.5 PROFIsafe über PROFINET

8.5.1 Grundlagen I-Device-F-Proxy

Kurzbeschreibung

Mithilfe des I-Device-F-Proxys können Sie eine PROFIsafe-Projektierung mit F-Host (F-CPU SIMATIC) an PROFINET mit SIMOTION Geräten (SIMOTION D, SIMOTION P, SIMOTION C) für die unterlagerten Antriebe erstellen. Das Routing zyklischer PROFIsafe Daten zu SINAMICS Integrated und SINAMICS Antriebe am externen PROFIBUS oder PROFINET ist damit möglich.

Ein F-Host kommuniziert über die I-Device Schnittstelle und einen F-Proxy einer SIMOTION CPU mit den Antrieben, die sich an PROFIBUS DP extern, PROFIBUS DP integrated und am PROFINET IO System der SIMOTION CPU befinden können. An den Kommunikationssträngen der SIMOTION CPU befinden sich SINAMICS S120/S110, incl. SINAMICS integrated/CX32/CX32-2 und G120.

Das überlagerte Projekt mit den F-CPUs ist das Master-Projekt. Im Master-Projekt kann es mehrere F-CPUs geben. Die unterlagerten Projekte mit SIMOTION CPUs werden als Sub-Projekte bezeichnet. Alle F-Slaves von allen Strängen (PROFIBUS/PROFINET), die von SIMOTION ausgehen, werden über den F-Proxy als n F-Submodule in einem F-Proxy Modul abgebildet.

Bei der Projektierung wird ein F-I-Device als Stellvertreter eines SIMOTION Geräts mit Antrieben erstellt und im Masterprojekt der F-CPU importiert.

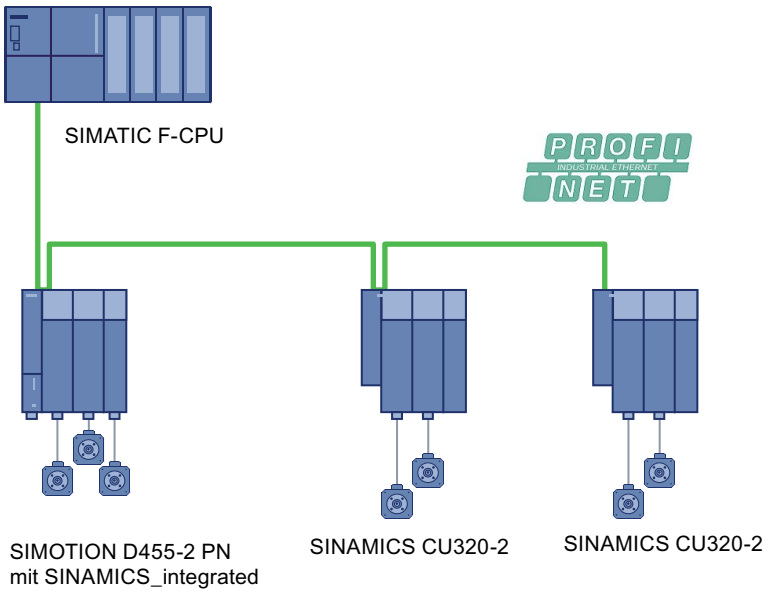


Bild 8-5 Beispieltopologie F-CPU Master mit SIMOTION D und unterlagerten Antrieben

Modul-/Submodulstruktur

Die Submodule des Antriebs mit dem Safety Telegramm werden auf Submodule der PROFINET I-Device Schnittstelle abgebildet, unabhängig davon, ob der Antrieb über PROFIBUS, integriertem PROFIBUS oder PROFINET an SIMOTION angebunden ist.

Beim I-Device-F-Proxy werden alle F-Proxy Submodule eines SIMOTION Gerätes in einem einzigen Modul (Modul 2) abgebildet.

Im folgenden Bild sehen Sie eine F-CPU als Master, dem eine SIMOTION mit integrierten und unterlagerten Antrieben untergeordnet ist.

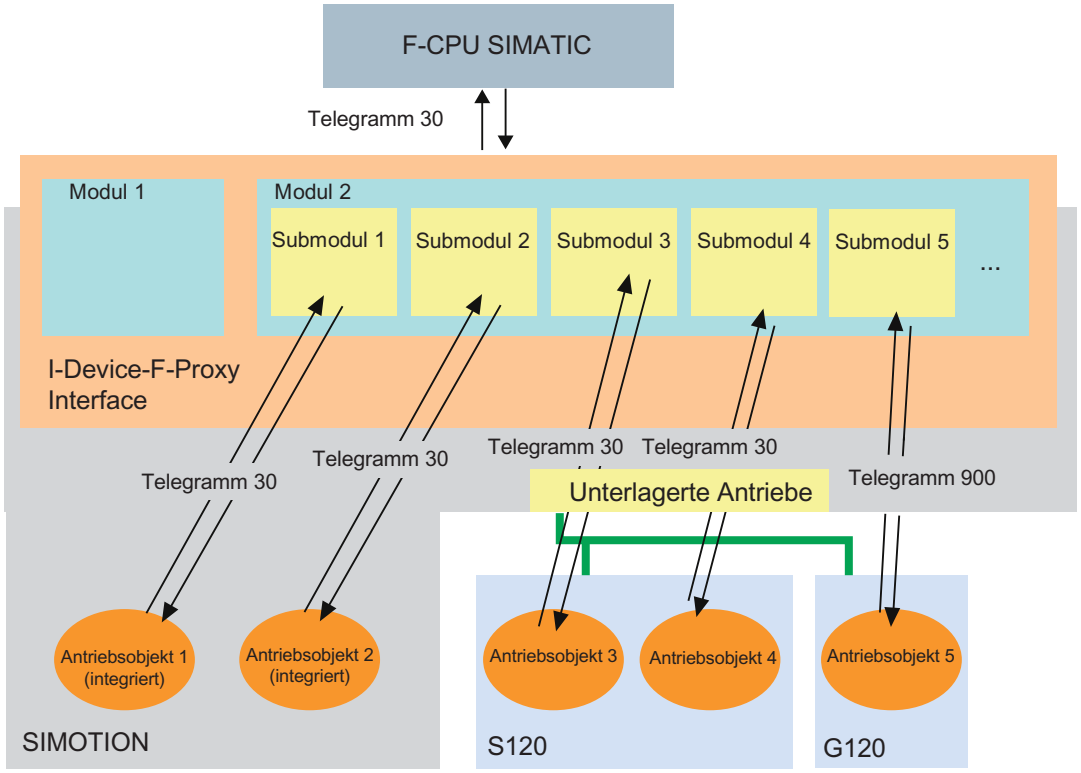


Bild 8-6 Master-Projekt mit einem Sub-Projekt als I-Device-F-Proxy

8.5.2 Unterstützte Geräte und Softwarevoraussetzungen I-Device-F-Proxy

Softwarevoraussetzung

- Step7 ab V5.5
- SINAMICS S120 ab V2.6
- SINAMICS G120 ab V3.2
- SIMOTION ab V4.2
- S7 F Configuration Pack ab V5.5 SP7 (bei Verwendung von Safety/PROFIsafe)
- Gültig ab PROFIsafe V2

Unterstützte Geräte

Antriebsgeräte

- SINAMICS Integrated
- SINAMICS Controller Extension (CX32, CX32-2)
- SINAMICS Geräte am unterlagerten PROFIBUS
- SINAMICS Geräte am unterlagerten PROFINET

Devices mit I-Device-F-Proxy Funktionalität

- SIMOTION D4xx, D4x5-2
- SIMOTION C240 PN
- SIMOTION P350-3, P320-3
- F-CPU ab V3.2

Tabelle 8- 1 Die Tabelle zeigt welche SIMOTION Geräten den I-Device-F-Proxy unterstützen und welche SINAMICS Geräte über den I-Device-F-Proxy erreichbar sind.

Übersicht der I-Device-F-Proxy fähigen Geräte	
SIMOTION IO-Controller V4.2	
Controller Based	C240 PN
PC Based	P320-3 P350-3
Drive Based (blocksize)	D410 PN
Drive Based (booksize)	D425 D435 D445 D445-1 D445-2 DP/PN D455-2 DP/PN
SINAMICS IO-Devices und DP-Slaves	
S120 CX32	CX32 CX32-2

Übersicht der I-Device-F-Proxy fähigen Geräte	
S120	CU320 DP CU320 CBE20 CU320-2 DP CU320-2 DP CBE20 CU310-2 DP CU310 DP CU310 PN CU310-2 PN CU320-2 PN
S110	CU305 CU305 PN
G120	CU240S PN F CU240D PN F

8.5.3 Detailbeschreibung/Eigenschaften I-Device-F-Proxy

Spezifische Eigenschaften des I-Device-F-Proxy

Azyklische Dienste am I-Device-F-Proxy

Das aktuelle I-Device Interface unterstützt keinen azyklischen Datentransfer. Die I-Device-F-Proxy Submodule haben keinen Parameterzugriff (Parameter Access Point PAP) und können keine Alarme durchleiten.

Falls ein Alarmkanal genutzt werden soll, muss der Antrieb an PROFINET betrieben werden und über Shared Device an SIMOTION und F-CPU direkt eingebunden sein. Dabei werden die F-Daten von der F-CPU und die Motion Control Daten von der SIMOTION verwaltet.

Unterstützte Telegramme

Generell werden alle PROFIsafe Telegramme unterstützt. In V4.2 ist dies nur Telegramm 30 bzw. Telegramm 900 für SINAMICS G120.

Taktsynchronisation

Die F-Proxy Submodule an der SIMOTION CPU sind RT und können nicht-taktsynchron betrieben werden.

Unterstützte unterlagerte Bussysteme

Unterstützt werden Antriebe am PROFIBUS integrated, am PROFIBUS extern und am PROFINET extern. Alle diese Busschnittstellen können auch gleichzeitig über den I-Device-F-Proxy geroutet werden.

Unterstützte I-Device Interfaces

Es werden alle PROFINET fähigen SIMOTION Geräte unterstützt.

Unterstützte Anzahl von F-Proxy-Submodulen

Unterstützt werden maximal 128 I-Device Submodule. Davon sind bis zu 64 Submodule für Safety nutzbar. Die weiteren 64 Submodule stehen für Standard IOs zur Verfügung.

Reaktionszeiten - Durchleitungszeit I-Device-F-Proxy

Werden Daten von der F-CPU über den I-Device-F-Proxy übertragen, verlängert sich die Laufzeit zu den SINAMICS-Antrieben durch den I-Device-F-Proxy um maximal 2 Servo-Takte pro Senderichtung. Als Servo-Takt verwenden Sie den Servo-Takt des unterlagerten Systems.

8.5.4 Topologieübersichten I-Device-F-Proxy

8.5.4.1 Topologie I-Device-F-Proxy für PROFIBUS Antriebsgeräte

Topologiebeispiel

Im folgenden Bild sehen Sie eine schematische Topologiedarstellung, bei der die Safety Antriebe über PROFIBUS DP an die SIMOTION CPU angeschlossen sind.

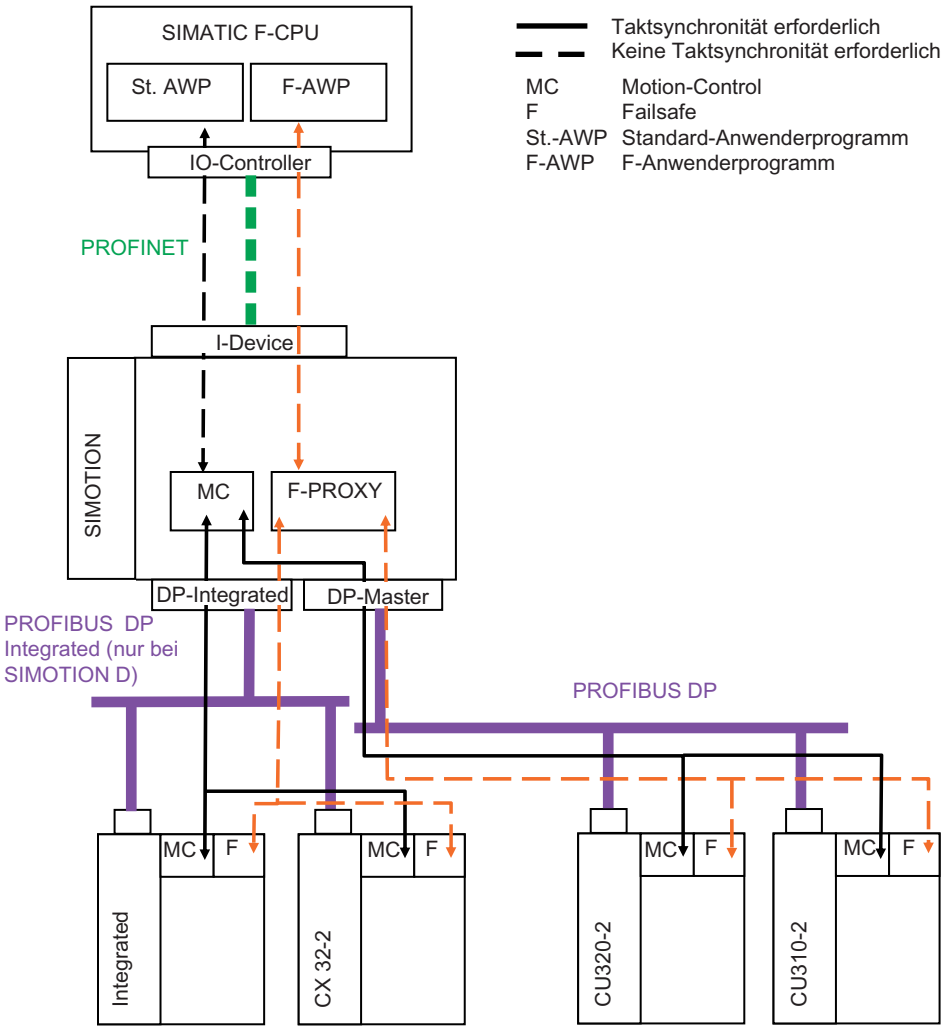


Bild 8-7 Topologie I-Device-F-Proxy für PROFIBUS Antriebsgeräte

8.5.4.2 Topologie I-Device-F-Proxy für PROFINET Antriebsgeräte

Topologiebeispiel

Im folgenden Bild sehen Sie eine schematische Topologiedarstellung, bei der die Safety Antriebe über PROFINET bzw. die internen über PROFIBUS DP Integrated an die SIMOTION CPU angeschlossen sind.

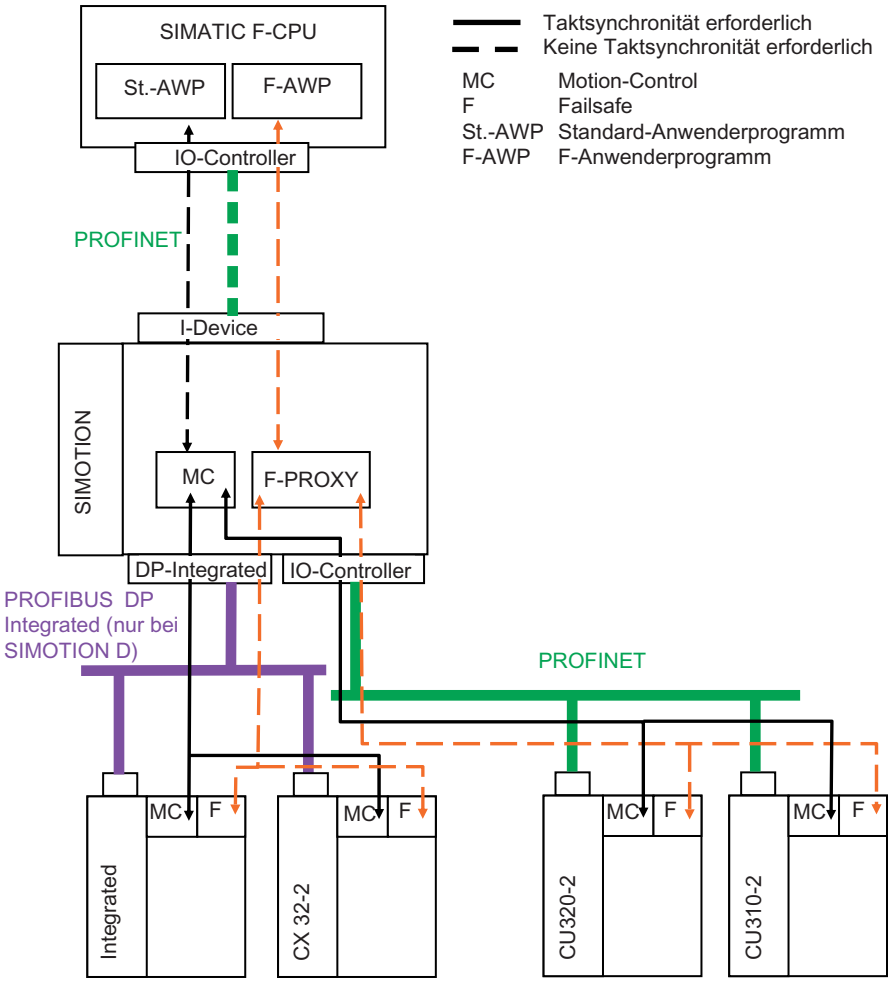


Bild 8-8 Topologie I-Device-F-Proxy für PROFINET Antriebsgeräte

8.5.4.3 Topologie I-Device-F-Proxy für PROFIBUS und PROFINET Antriebsgeräte

Topologiebeispiel

Im folgenden Bild sehen Sie eine schematische Topologiedarstellung, bei der die Safety Antriebe über PROFINET und PROFIBUS DP an die SIMOTION CPU angeschlossen sind.

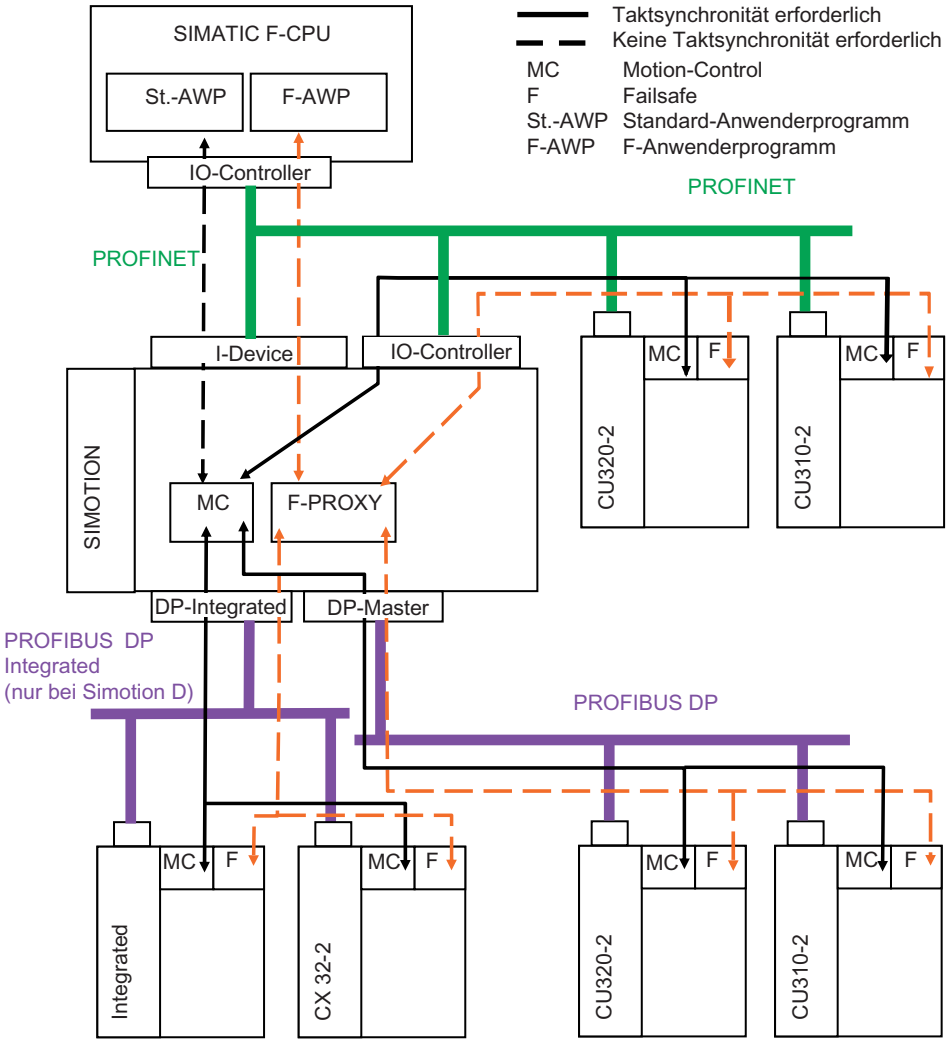


Bild 8-9 Topologie I-Device-F-Proxy für PROFIBUS und PROFINET Antriebsgeräte

8.5.5 Projektieren I-Device-F-Proxy

8.5.5.1 Prinzipieller Projektierungsablauf I-Device-F-Proxy

Projektierungsanforderungen

Ein F-Host kommuniziert über die I-Device Schnittstelle und einen F-Proxy einer SIMOTION CPU mit den Antrieben, die sich an PROFIBUS DP extern, PROFIBUS DP integrated und am PROFINET IO System der SIMOTION CPU befinden können.

Grundlegender Projektierungsablauf

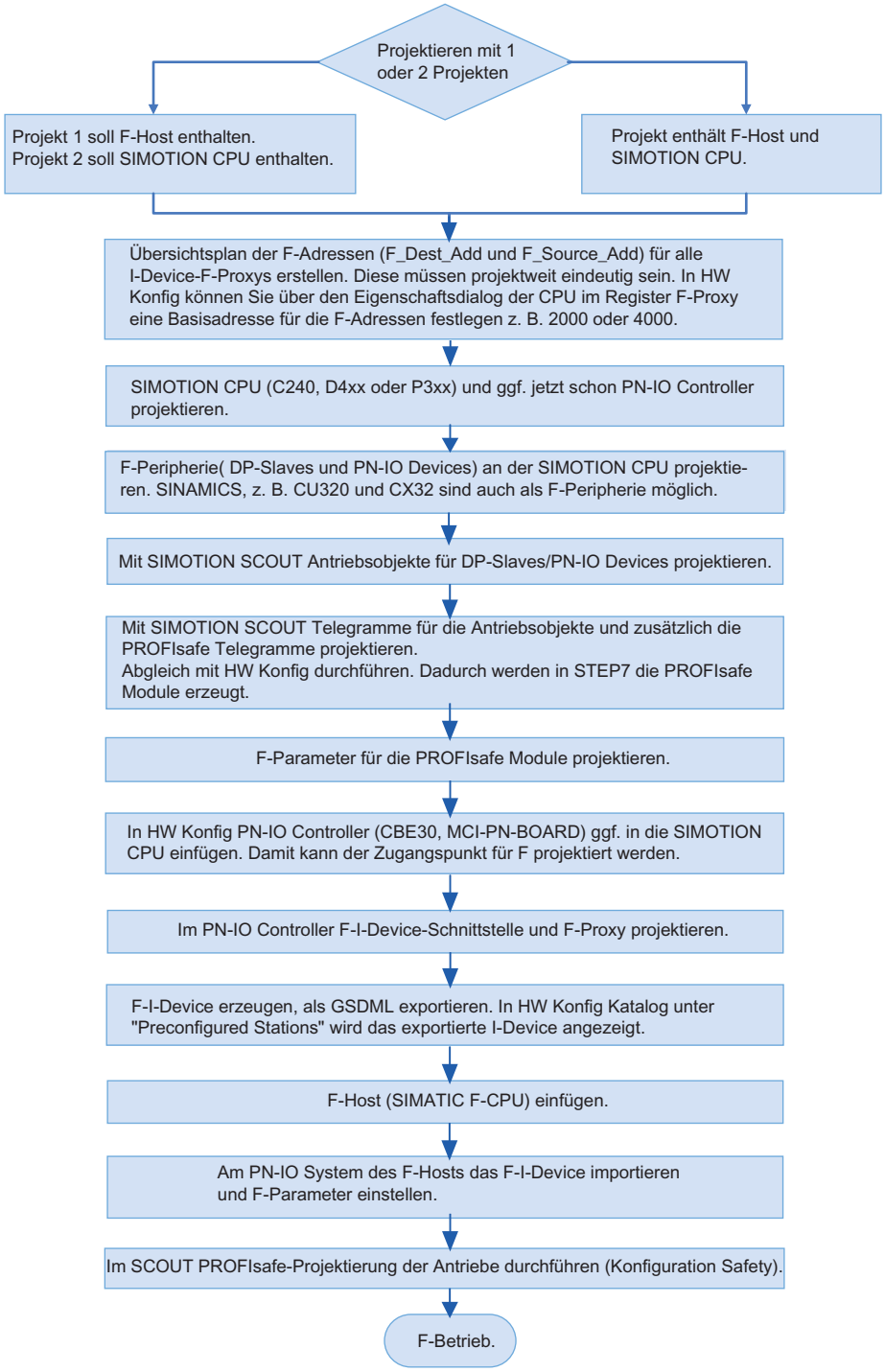


Bild 8-10 Prinzipieller Projektierungsablauf I-Device-F-Proxy

8.5.5.2 Projektierungsbeispiel SIMOTION D435 und SINAMICS S120 über PROFINET

Projektierungsbeispiel I-Device-F-Proxy

Im folgenden Beispiel projektieren Sie eine SIMATIC F-CPU 317F-2PN/DP V3.2. Des Weiteren verwenden Sie eine SIMOTION D435 mit einem SINAMICS S120 CU320 PN, der über die PN-Schnittstelle (CBE30) mit der SIMOTION verbunden ist. Der exportierte I-Device-F-Proxy der unterlagerten SIMOTION CPU wird dann in die überlagerte F-CPU importiert. Im Beispiel wird die Projektierung mit nur einem Projekt dargestellt.

So projektieren Sie einen I-Device-F-Proxy

1. Erstellen Sie einen Übersichtsplan mit der F-CPU, der SIMOTION CPU und den Antrieben, die PROFIsafe unterstützen sollen. Im Beispiel sind dies nur eine SIMOTION CPU und ein SINAMICS S120. Tragen Sie in diesen Übersichtsplan die Basisadressen der CPU und der erforderlichen Antriebe ein. Die festgelegten Adressen sollten Sie später bei der Projektierung verwenden. Der Übersichtsplan ist nur für größere Projekte notwendig.
2. Erstellen Sie in SIMOTION SCOUT ein neues Projekt.

   Weitere Informationen zur Projektierung mit PROFINET finden Sie im Kapitel PROFINET IO mit SIMOTION projektieren (Seite 75).
3. Fügen Sie eine SIMOTION D435 V4.2 ein. Die Checkbox **HW Konfig öffnen** muss aktiviert sein. Bestätigen Sie mit **OK**. HW Konfig wird geöffnet.
4. Tragen Sie bei der SIMOTION CPU in HW Konfig eine Basisadresse für die F-Adressen ein. Alle F_Dest_Add für die unterlagerten Antriebe starten dann mit dieser Basis und sind bei umfangreichen Projekten leichter zur verwalten. Verwenden Sie als Basis z. B. 4000, wird die erste F_Dest_Add des Antriebs mit 400 vergeben usw. Standardmäßig ist die Basis 2000.
5. Fügen Sie ggf. ein CBE30 ein und projektieren Sie das PROFINET-Netz.
6. Fügen Sie einen SINAMICS S120 CU320 PN (ab V2.6) in HW Konfig ein und projektieren Sie die Schnittstelle.
7. Konfigurieren Sie das Antriebsgerät in SIMOTION SCOUT mithilfe des Assistenten.
8. Fügen Sie daher ein neues TO Achse ein und durchlaufen Sie den Achsassistenten. Im Assistent verschalten Sie die Achse auf das entsprechende Antriebsobjekt des S120 und automatisch wird ein entsprechendes Telegramm angelegt (Symbolische Zuordnung verwenden).
9. Speichern und übersetzen Sie das Projekt.
10. Nach der Konfiguration müssen Sie das PROFIsafe Telegramm anwählen. Doppelklicken Sie im Projektnavigator von SIMOTION SCOUT unterhalb des Antriebsgerätes auf **<"Antriebsgerät_xx"> - Kommunikation > Telegrammkonfiguration**. Im Arbeitsbereich werden die Telegramme aufgeblendet.

11. Markieren Sie den Antrieb im Register **IF1: PROFIdrive PZD-Telegramm** der Telegrammübersicht und wählen Sie im unteren Bereich des Fensters unter **Telegrammkonfiguration anpassen** den Eintrag **PROFIsafe hinzufügen**. Das PROFIsafe Telegramm wird eingefügt.

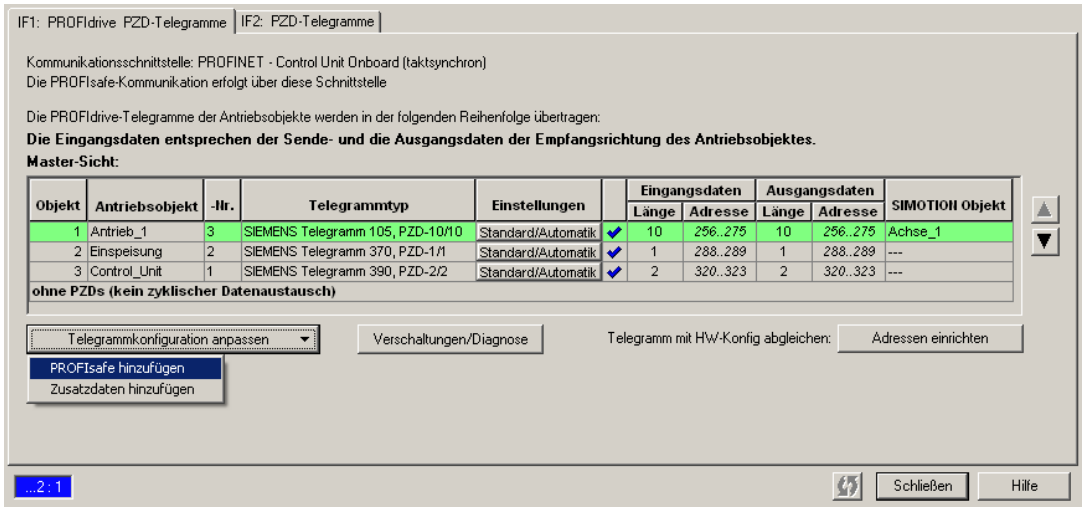


Bild 8-11 PROFIsafe Telegramm einfügen

12. Speichern und übersetzen Sie das Projekt.
13. Klicken Sie auf **Adresse einrichten**, um einen Abgleich zwischen SIMOTION und HW Konfig durchzuführen. Mit dem Speichern ist die Projektierung im SIMOTION SCOUT abgeschlossen.
14. Wechseln Sie in HW Konfig und projektieren Sie SYNC Master und Slave und eine taktsynchrone Applikation.
15. Doppelklicken Sie in der Stationsübersicht SIMOTION D435 auf X1400 P1 und wählen Sie im Register **Topologie** unter **Partner-Port** die Belegung vom Antriebsgerät. Bestätigen Sie mit **OK**.

16. Doppelklicken Sie in der Station auf die PN IO-Schnittstelle um in den Eigenschaften den I-Device Modus zu aktivieren. Aktivieren Sie im Register **I-Device** die Checkbox **I-Device Modus**. Klicken Sie auf **Neu...** und wählen Sie im aufgeblendeten Dialog unter **Transferbereichstyp** den Eintrag **F-Peripherie**.

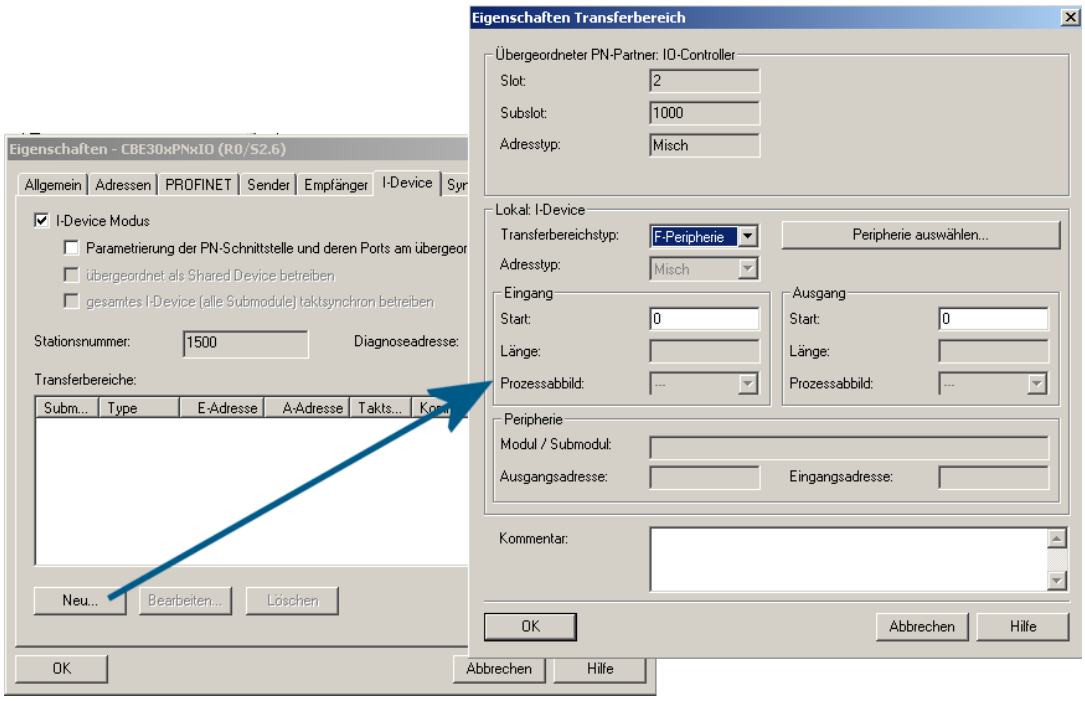


Bild 8-12 Eigenschaften I-Device Transferbereich

17. Klicken Sie daneben auf den Button **Peripherie auswählen** und wählen Sie im aufgeblendeten Dialog den entsprechenden PROFIsafe Kanal aus. Bestätigen Sie beide Dialoge mit **OK**.
Damit ist die Konfiguration des unterlagerten Antriebs abgeschlossen. Sie müssen nur noch die Station speichern und übersetzen.

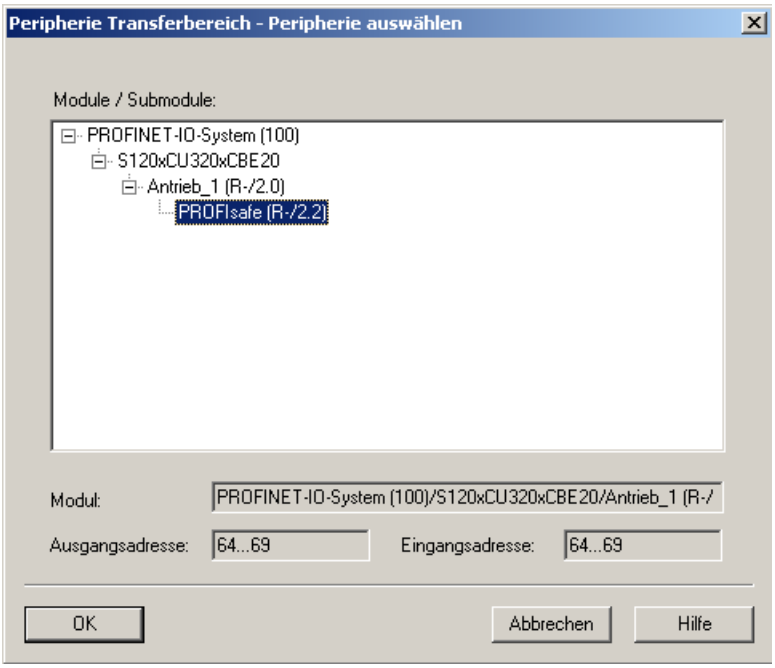


Bild 8-13 PROFIsafe Kanal der Peripherie wählen

18. Die PROFIsafe Einstellungen können Sie beim Antriebsgerät einsehen. In der Detailansicht des Baugruppenträgers finden Sie den Eintrag **PROFIsafe**. Doppelklicken Sie darauf, um die Eigenschaften anzuzeigen.

    Im Register **PROFIsafe** finden Sie die F-Adresse unter **F_Dest_Add**. Diese Adresse muss im Gesamtprojekt eindeutig sein. Falls Sie mehrere F-Proxy verwenden, müssen Sie dafür sorgen, dass diese Adresse nur einmal vergeben wird. Ändern Sie bei Bedarf diesen Wert. Die F-Adresse wird in HW Konfig in der Detailansicht im Stationsfenster beim Kommentar angezeigt. Damit ist bei mehreren Teilnehmern eine übersichtliche Zuordnung möglich.

    Die Parameter F_CRC_Length=3-Byte-CRC und F_Par_Version=1 kennzeichnen den PROFIsafe V2-Mode. Achten Sie daher auf diese Werte, da nur ab dieser Version ein I-Device-F-Proxy Projektierung möglich ist.

Weitere Informationen zu den F-Parametern finden Sie unter Eigenschaften PROFIsafe bei der Projektierung (Seite 214)

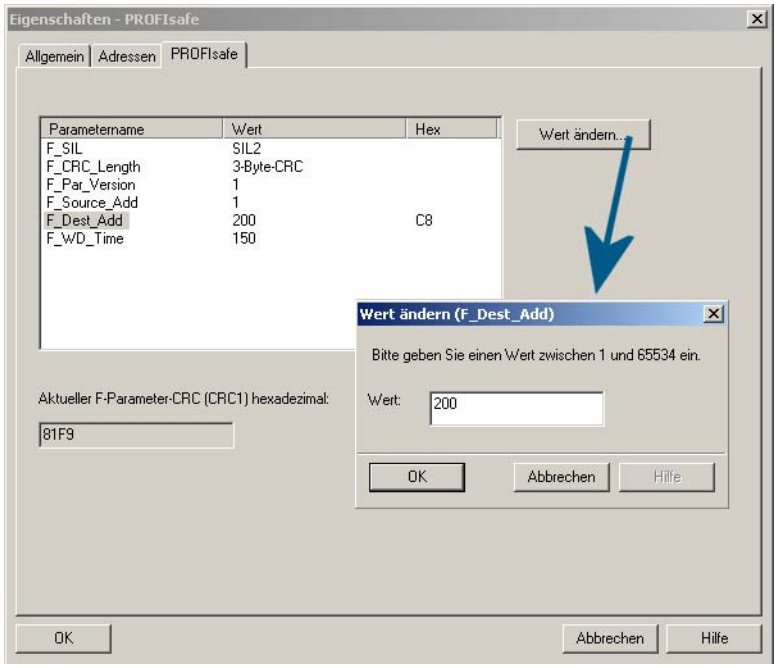


Bild 8-14 F-Adresse (F_dest_Add) einstellen

19. Die F-Adresse muss mit der PROFIsafe-Adresse bei der Safety Projektierung des Antriebs in SIMOTION SCOUT übereinstimmen. Im Beispiel ist dies die Adresse 200 dec bzw. C8H. Die Adresse tragen Sie bei der Konfiguration von Safety Integrated im Fenster Konfiguration ein. Der Wert wird in den Parametern p9610/p9810 des Antriebs hinterlegt.

Bild 8-15 F_Dest_Add als PROFIsafe Adresse eintragen

20. Erstellen Sie nun die GSD-Datei für den I-Device-F-Proxy. Wählen Sie im Menü **Extras > GSD-Datei für I-Device erstellen.** Klicken Sie im aufgeblendeten Dialog auf **Erstellen** und dann auf **Installieren**. Das I-Device wird unter **Preconfigured Stations** im Hardwarekatalog angezeigt.
21. Sie können nun ein neues Projekt mit F-CPU erstellen oder das bestehende Projekt im SIMATIC Manager öffnen und verwenden. Im Beispiel öffnen Sie das bestehende Projekt im SIMATIC Manager.
22. Wählen Sie im Menü z. B. **Einfügen > Station > SIMATIC 300-Station.** Doppelklicken Sie auf die Station und dann auf den Eintrag **Hardware**. HW Konfig wird geöffnet.

23. Aus dem Hardwarekatalog fügen Sie z. B. ein S7 300 Rack ein, wenn Sie eine F-CPU aus der S7 300 Reihe wählen wollen.
24. Fügen Sie die F-CPU z. B. CPU317-2 PN/DP ein. Diese muss mindestens die Version V3.2 besitzen.
25. Ziehen Sie das vorher erstellte I-Device unter **Preconfigured Stations** per Drag&Drop an das PROFINET IO Netz. Nach dem Speichern und Übersetzen ist die Projektierung abgeschlossen.

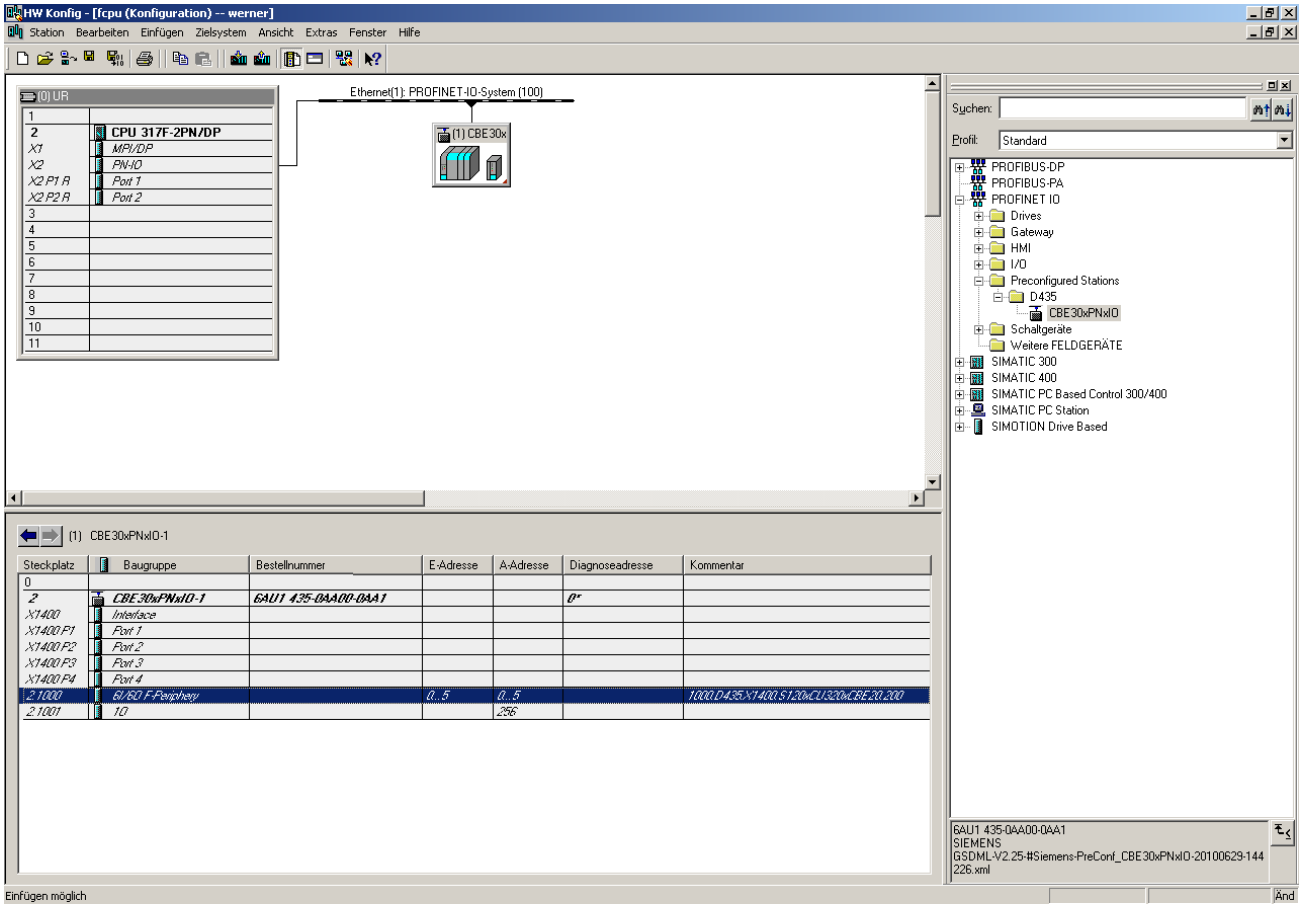


Bild 8-16 Master Projekt mit einem Submodul I-Device-F-Proxy

8.5.5.3 F-Adresse im bestehenden Projekt anpassen

F-Adresse (F_Dest_Add) projektweit anpassen

Bei einem bestehenden Projekt können Sie nachträglich prüfen, ob die F-Adresse für den I-Device-F-Proxy richtig eingestellt ist. Die F-Adresse muss an folgenden Stellen gleich sein:

- PROFIsafe Slot des Antriebs an der SIMOTION CPU (HW Konfig)
- PROFIsafe Slot des I-Device der SIMOTION CPU (HW Konfig)
- Safety Konfiguration beim Antrieb in SIMOTION (SIMOTION SCOUT)

Wenn Sie nachträglich die F-Adressen ändern wollen ohne das I-Device nochmals zu erstellen und zu installieren, müssen Sie die Änderung an den oben genannten drei Stellen vornehmen.

Im folgenden Beispiel stellen Sie bei einer SIMOTION CPU mit Antrieb am PROFINET und einer F-CPU in einem gemeinsamen Projekt die F-Adresse auf den Wert 300 (12CHex) ein.

So prüfen Sie, ob die F-Adresse identisch ist

1. Öffnen Sie in HW Konfig das Projekt der SIMOTION CPU.
2. Doppelklicken Sie in der Detailansicht des Antriebsgeräts auf **PROFIsafe**. Wechseln Sie im geöffneten Fenster in das Register **PROFIsafe**. Unter **F_Dest_Add** muss der Wert 300 stehen. Zum Ändern der **F_Dest_Add** klicken Sie auf den Button **Wert ändern** und tragen im aufgeblendeten Dialog 300 ein.

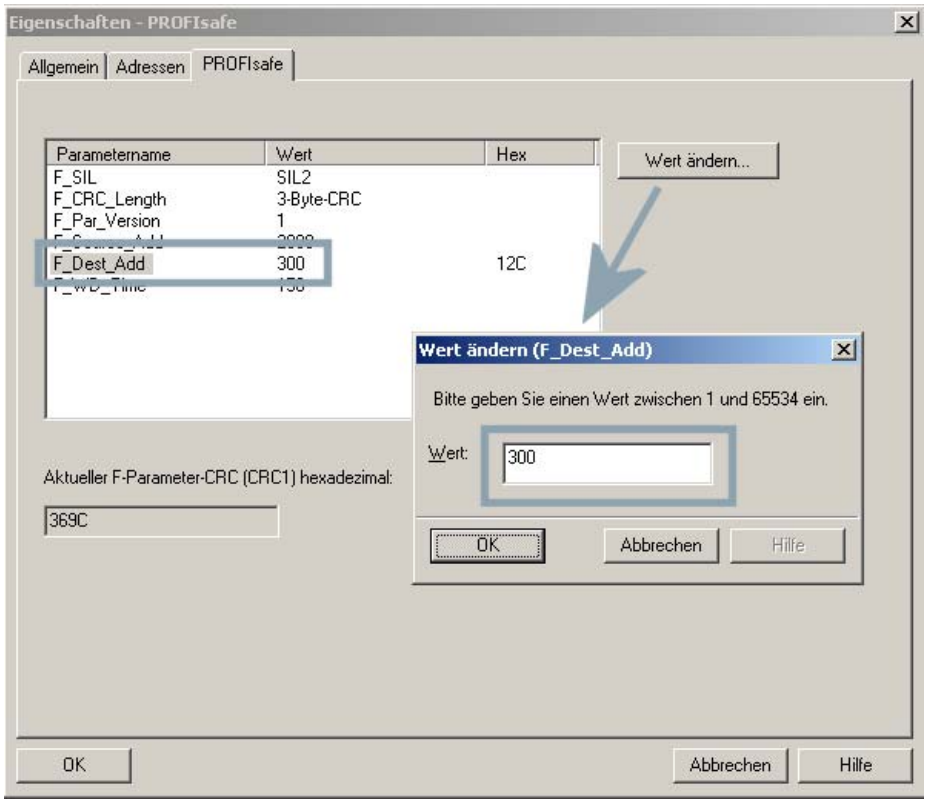


Bild 8-17 F-Adresse PROFIsafe Slot am Antriebsgerät

3. Bestätigen Sie mit **OK** und speichern und übersetzen Sie das Projekt.
4. Öffnen Sie das Projekt mit der F-CPU in HW Konfig.

5. Doppelklicken Sie in der Detailansicht des I-Devices auf die **PROFIsafe Peripherie** z. B. 6I/6O F-Periphery. Wechseln Sie im geöffneten Fenster in das Register **PROFIsafe**. Unter **F_Dest_Add** muss der Wert 300 stehen. Zum Ändern der **F_Dest_Add** klicken Sie auf den Button **Wert ändern** und tragen im aufgeblendeten Dialog 300 ein.

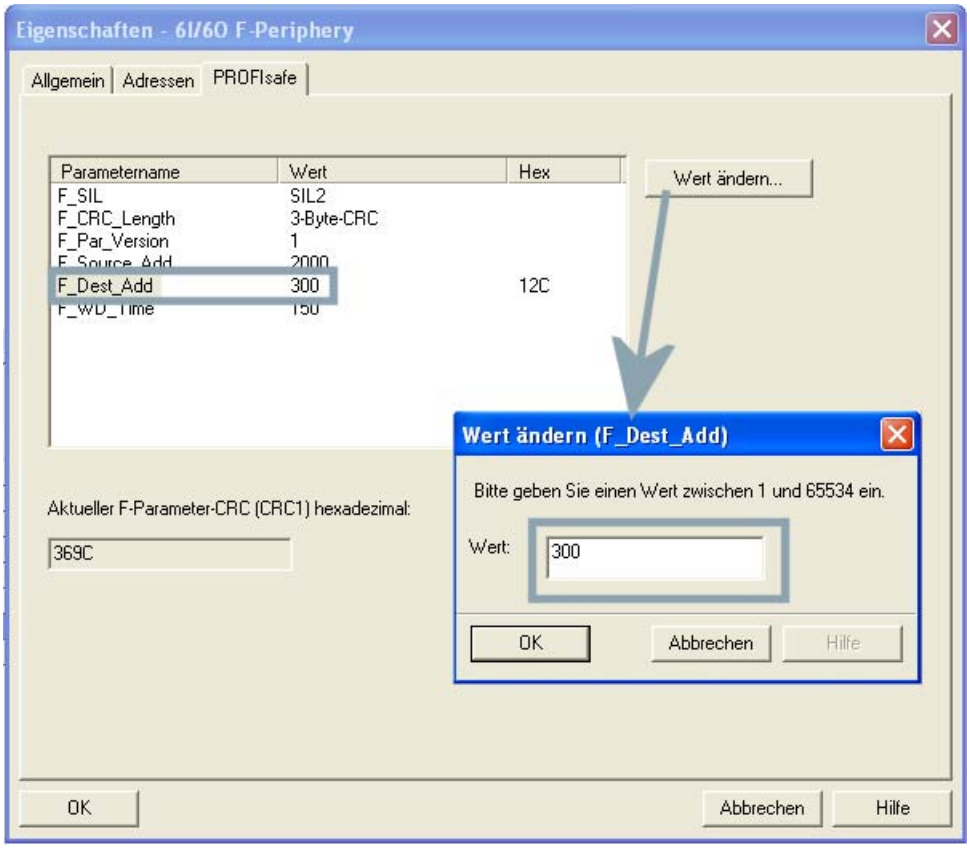


Bild 8-18 F-Adresse beim I-Device an F-CPU

6. Bestätigen Sie mit **OK** und speichern und übersetzen Sie das Projekt.
7. Öffnen Sie das SIMOTION Projekt in SIMOTION SCOUT.
8. Navigieren Sie im Projektnavigator zum Antrieb z. B. (D435 > S120xCU320xCBE20 > Antriebe > Antrieb_1).
9. Doppelklicken Sie im Projektnavigator unter Funktionen auf **Safety Integrated**.
10. Klicken Sie auf den Button **Konfiguration**. Das Fenster Konfiguration wird geöffnet.

11. Prüfen Sie den Wert 12CHex (300) unter **PROFIsafe-Adresse** und ändern diese falls nötig.

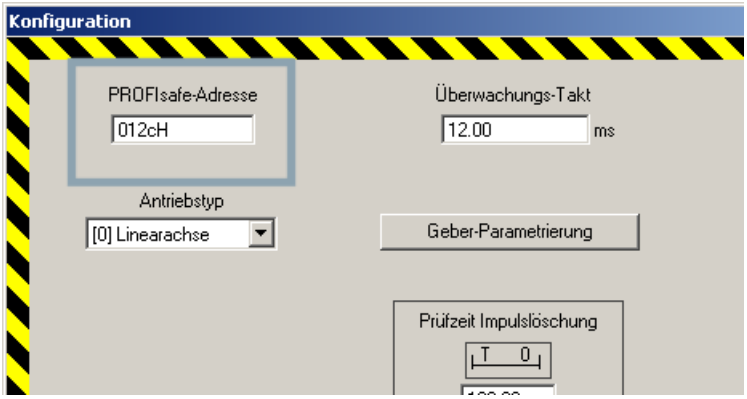


Bild 8-19 F-Adresse Konfiguration Safety Integrated

12. Bestätigen Sie mit **Schließen** und speichern und übersetzen Sie das Projekt.

8.5.5.4 Projektierung D435 mit S120 an PROFINET und Integriertem PROFIBUS

Integrierten Antrieb an D435 für PROFIsafe

Im vorherigen Beispiel haben Sie einen S120 an D435 mit PROFINET projektiert und als I-Device importiert. Nun wird das Projekt um einen SINAMICS_Integrated über den internen PROFIBUS DP ergänzt.

So projektieren Sie einen SINAMICS_Integrated

Sie haben ein Projekt mit D435 und SINAMICS S120 projektiert und als I-Device an eine F-CPU importiert.

1. Konfigurieren Sie in SIMOTION SCOUT das Antriebsgerät am SINAMICS_Integrated und fügen Sie ein PROFIsafe Telegramm ein (Siehe Punkte 7-10, Beispiel Projektierungsbeispiel SIMOTION D435 und SINAMICS S120 über PROFINET (Seite 226)).
2. Markieren Sie in HW Konfig den SINAMICS_Integrated und doppelklicken Sie in der Detailansicht des Baugruppenträgers auf das PROFIsafe Module.
3. Im Register **Konfiguration** des aufgeblendeten Dialogs markieren Sie **PROFIsafe Telegramm** und klicken auf den Button **PROFIsafe...** . Im Dialog **PROFIsafe Eigenschaften** können Sie die F-Parameter einsehen und ändern.
   (Falls der Button **PROFIsafe...** nicht angezeigt wird, müssen Sie erst den Button **Aktivieren** klicken, um Änderungen vorzunehmen.)

4. Doppelklicken Sie die PN IO Schnittstelle der SIMOTION CPU. Aktivieren Sie im aufgeblendeten Dialog im Register **I-Device** die Checkbox **I-Device Modus**. Klicken Sie auf **Neu...** und wählen Sie im aufgeblendeten Dialog unter **Transferbereichstyp** den Eintrag **F-Peripherie**. Klicken Sie auf den Button **Peripherie auswählen** und wählen Sie im aufgeblendeten Dialog den entsprechenden PROFIsafe Kanal aus.

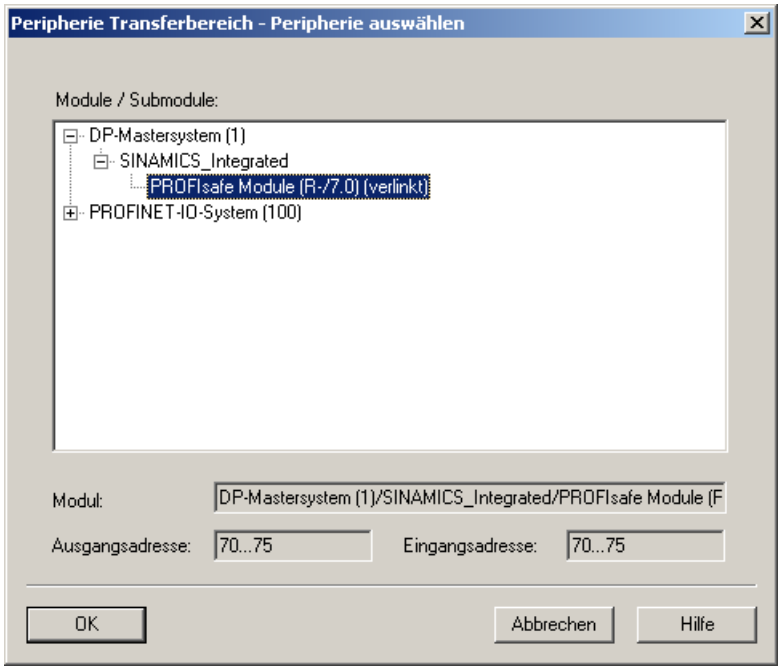


Bild 8-20 PROFIsafe Modul für SINAMCIS_Integrated für Transferbereich frei geben

5. Bestätigen Sie den Dialog mit **OK**. Im Dialog Eigenschaften Transferbereich werden die Ein-/Ausgänge zugeordnet und ein automatisch generierter Kommentar angezeigt. Dieser Kommentar enthält u. a. den Subslot, den SIMOTION Gerätenamen, den Anschluss und Gerätenamen des Antriebs. Am Ende steht die F_Dest_Add. Den Kommentar können Sie ggf. ändern.

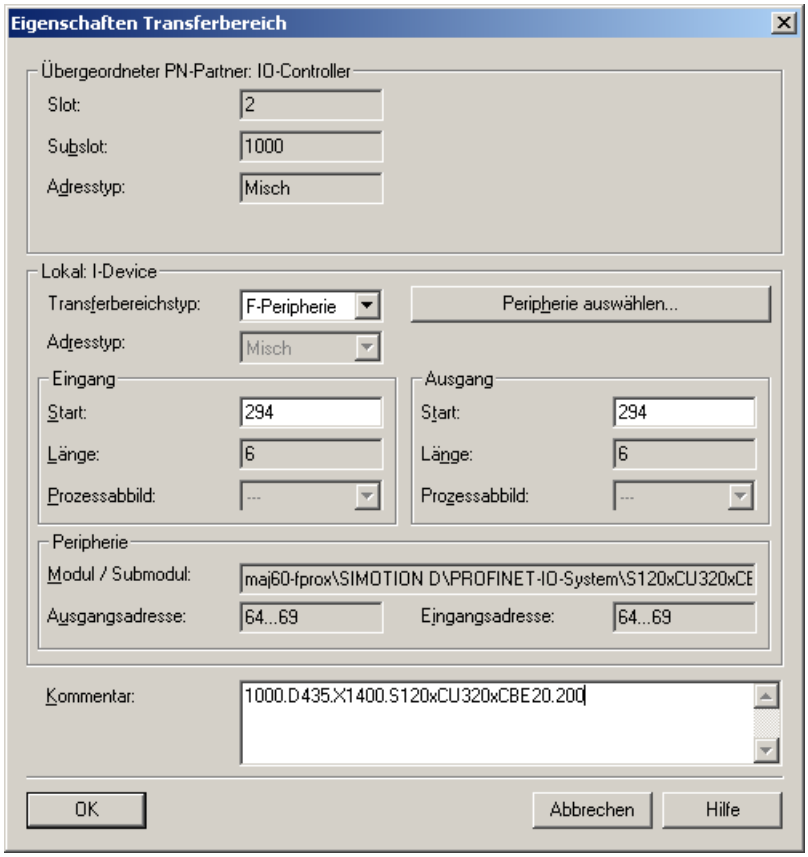


Bild 8-21 Eigenschaften I-Device Transferbereich Kommentar

6. Bestätigen Sie diesen Dialog mit **OK**. Im Transferbereich werden die F-Daten der beiden Antriebe angezeigt.
   Damit ist die Konfiguration des unterlagerten Antriebs abgeschlossen. Sie müssen nur noch die Station speichern und übersetzen.

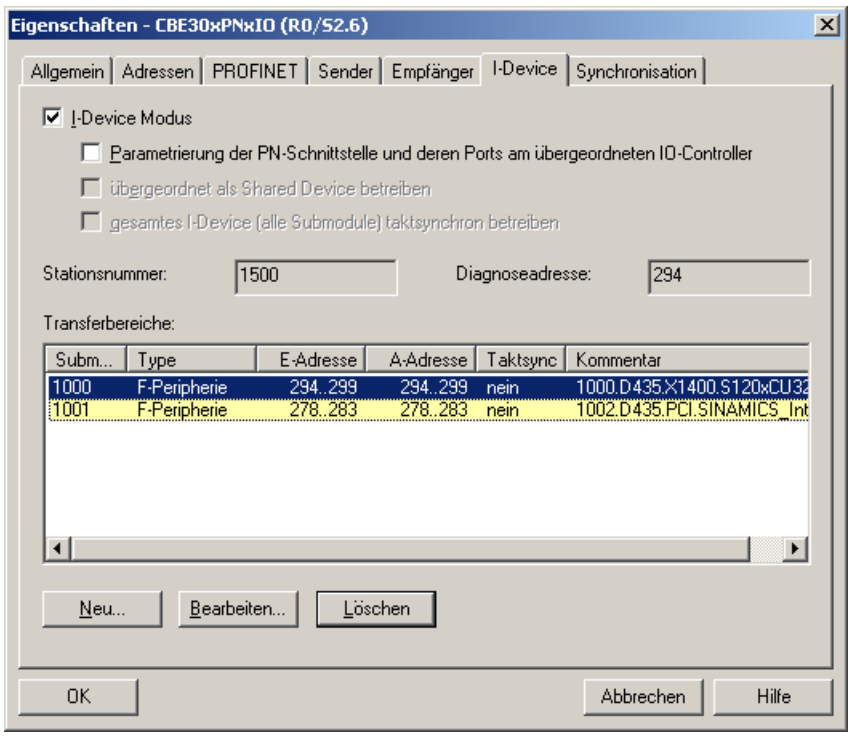


Bild 8-22 Transferbereich I-Device

7. Erstellen Sie nun die GSD-Datei für den I-Device-F-Proxy. Wählen Sie im Menü **Extras > GSD-Datei für I-Device erstellen**. Klicken Sie im aufgeblendeten Dialog auf **Erstellen** und dann auf **Installieren**. Das I-Device wird unter Preconfigured Stations im Hardwarekatalog angezeigt.
8. Erstellen Sie wie im vorherigen Beispiel beschrieben im Projekt eine F-CPU und fügen Sie das I-Device des SIMOTION Moduls ein.
   Im Bild sehen Sie das Projekt mit F-CPU und I-Device-F-Proxy mit einem Antrieb an PROFINET und einem Antrieb am SINAMICS_Integrated an einer D435.

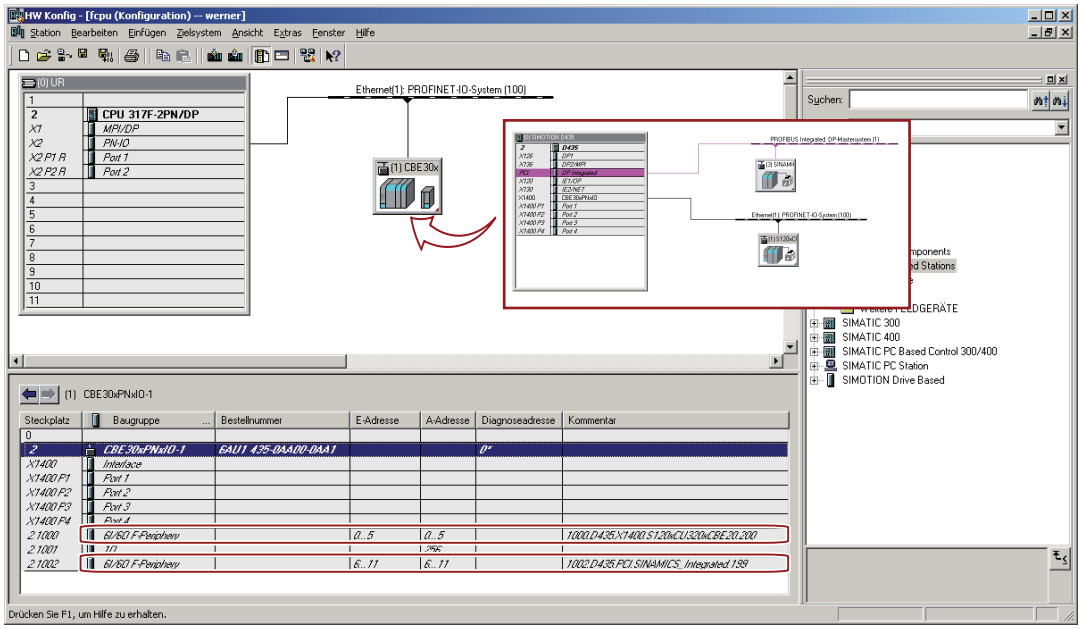


Bild 8-23 F-CPU mit I-Device-F-Proxy an PROFINET und integriertem PROFIBUS

8.5.5.5 Bestehende Anlage mit PROFIsafe über PROFIBUS auf PROFIsafe über PROFINET umrüsten

PROFIBUS auf PROFINET

Soll bei einer bestehenden Anlage die PROFIsafe Kommunikation, die vorher über PROFIBUS betrieben wurde, auf PROFINET umgestellt werden, muss die Anlage umgerüstet werden.

So gehen Sie vor bei der Umrüstung.

1. Löschen Sie die alte I-Slave Kopplung.
2. Stellen Sie ggf. die DP-Schnittstelle von DP-Slave auf DP-Master um (SIMOTION CPU).
3. Die PROFIsafe Telegrammprojektierung für die Antriebe kann unverändert bleiben.

4. Stellen Sie ggf. die F-Parameter **F_Par_Version = 1** und **F_CRC_Length = 3-Byte-CRC** ein, um den PROFIsafe V2 Standard zu verwenden. Sonst ist eine Kopplung im I-Device nicht möglich. Beim neu Anlegen von PROFIsafe Slots wird automatisch der V2 ausgewählt.

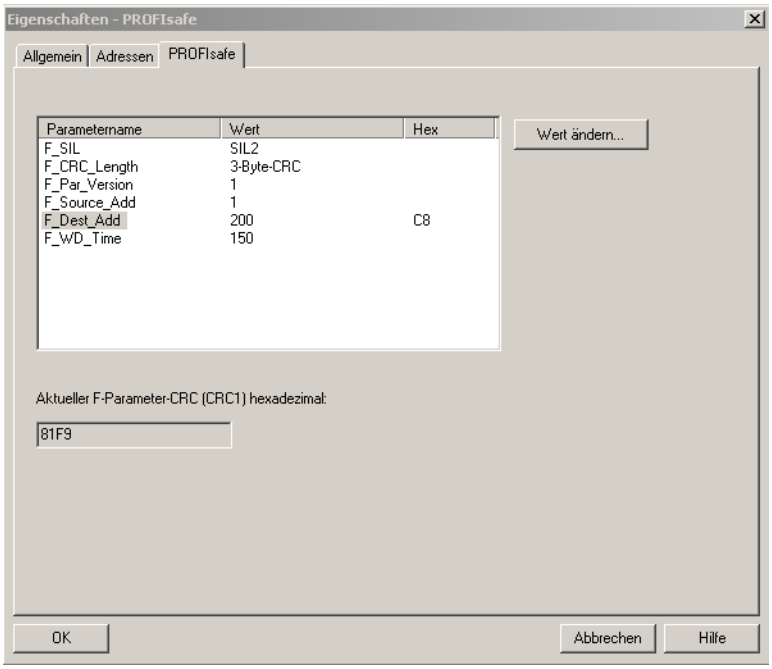


Bild 8-24 Eigenschaften PROFIsafe am Beispiel I-Device-F-Proxy

5. Wählen Sie den I-Device Modus in der HW Konfig auf der CBE30 an.
6. Wählen Sie über **Neu** und im Transferbereichstyp **F-Peripherie** die Peripherie aus und legen somit ein Submodul an.
7. Anschließend erstellen Sie über **Extras > GSD-Datei für I-Device erstellen...** eine neue GSD und installieren diese.
8. Koppeln Sie die GSD-Datei an die S7 F-CPU.

8.5.5.6 Allgemeines zu F-Adressen beim I-Device-F-Proxy

Kommunikationsadressen F-Source- bzw. F_Destination-Address

Legen Sie neue F-Hosts, F-Module bzw. F-Submodule im HW Konfig an, dann macht HW Konfig einen Vorschlag für die F-Source bzw. F-Destination Adresse als Defaulteinstellung. Diese Defaulteinstellung können Sie ändern bzw. überschreiben. Grundlage für die Default F-Adressvergabe ist der Parameter F-Basisadresse an der F-CPU bzw. auch an der SIMOTION CPU.

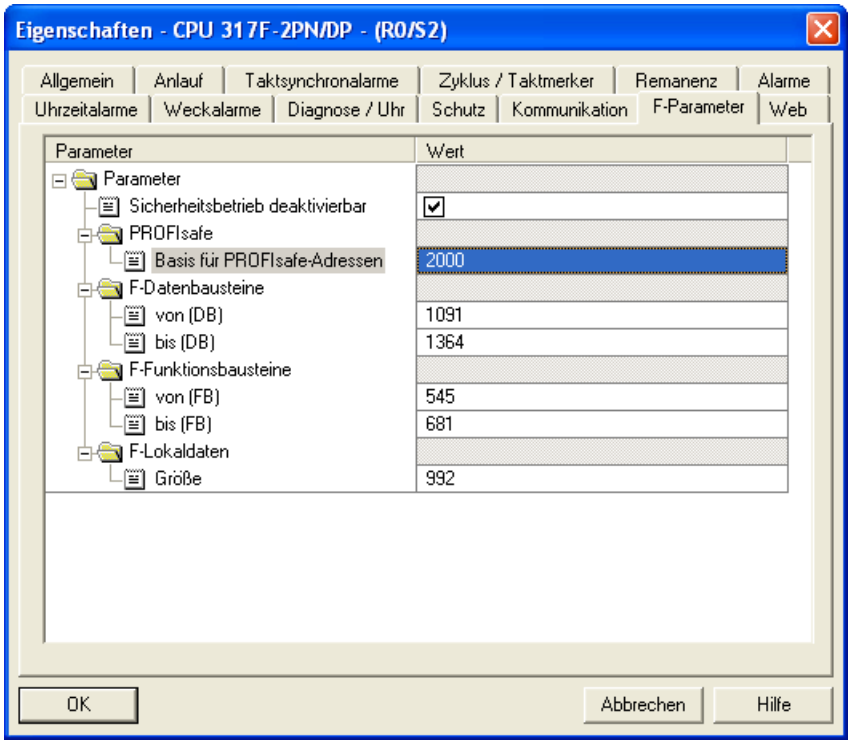


Bild 8-25 PROFIsafe Basisadresse

Die F_Source_Address wird nach demselben Verfahren vergeben wie bei Siemens F-Baugruppen mit OM:

- PROFIsafe-Basisadresse der CPU + Nummer des DP-Mastersystems bei PROFIBUS
- PROFIsafe-Basisadresse der CPU bei PROFINET
- 1, wenn die CPU keinen Parameter PROFIsafe-Basisadresse hat (Standard-CPU und F-H-CPU)

Richtlinien für die Adressvergabe

- Bei der Adressvergabe wird der volle Wertebereich von 1...65534 genutzt.
- Die Vergabe der F_Destination_Address erfolgt automatisch.
- Die automatische Vergabe erfolgt beim Stecken analog zu den anderen F-Baugruppen in Step7: ausgehend von einem Startwert wird absteigend nach der nächsten freien Adresse gesucht.

- Der Startwert ist die PROFIsafe-Basisadresse der CPU/10 (oder 1022, falls keine PROFIsafe-Basisadresse existiert oder ihr Wert größer 10000 ist).
- Erfolgt eine Änderung der F-Basisadresse (Parameter der CPU), dann werden bereits vergebene F-Source- und F-Dest-Adressen nicht nachgezogen (bleiben bestehen). Die Änderung wirkt sich nur auf die Defaultadresse von neu angelegten Submodulen aus.
- In einer F-CPU können mehrere F-Source Adressen vergeben werden. Die F-Source Adresse geht wie die F-Dest-Adresse in die CRC-Summe von PROFIsafe ein.
- Entfallen Adressen aus einem vergebenen Adressbereich (durch späteres Löschen von F-Submodulen), dann werden bei der Neuvergabe von Adressen zuerst die Lücken besetzt.

Hinweis

Wollen Sie mit Safety über die Projektgrenzen bzw. Proxies hinweg sinnvoll arbeiten, müssen Sie dafür sorgen, dass die F-Source- und F-Destination Adressen im Gesamtprojekt eindeutig vergeben werden. Vor Beginn der Projektierung sollten Sie ein F-Adressen-Plan haben/erstellen, in dem bestimmten Teilprojekten eindeutige F-Adressbereiche zugeordnet werden.

8.5.6 Shared Device über PROFINET

8.5.6.1 Allgemeines zum Shared Device

Beschreibung

Mit der neuen Funktionalität Shared Device können Sie über PROFINET den Zugriff auf ein IO-Device von mehreren IO-Controllern konfigurieren. Dies ermöglicht eine flexible Zuweisung von Kanälen/Modulen zu verschiedenen IO-Controllern und ist möglich für Ein- und Ausgänge. Einsetzen können Sie diesen Mechanismus, um z. B. auf die F-Daten eines Antriebs über die F-CPU zugreifen zu können, der unterhalb einer SIMOTION CPU projektiert ist.

Hinweis

Für die Projektierung von PROFIsafe wird die Projektierung mit der I-Device-F-Proxy Funktionalität empfohlen anstatt Shared Device.

Schematische Darstellung Shared Device

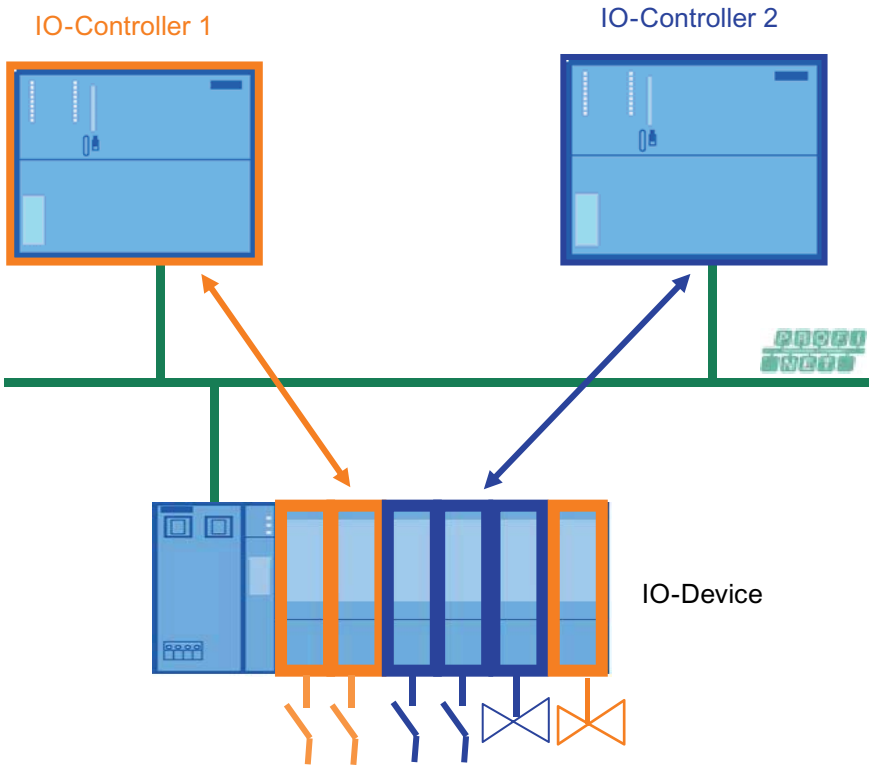


Bild 8-26 Schematische Darstellung Shared Device

Softwarevoraussetzungen

- SIMATIC Step 7 ab V5.5
- SINAMICS Firmware and Support Packages ab V4.3.2
- SIMOTION SCOUT ab V4.2
- S7 F Configuration Pack ab V5.5 SP7 (bei Verwendung von Safety/PROFIsafe)
- GSDML Datei ab V2.25

8.5.6.2 Shared Device in einem STEP 7 Projekt

Einleitung

Im folgenden Beispiel wird die einfachste Konfiguration eines Shared Devices beschrieben: Zwei IO-Controller (SIMOTION D445-2 DP/PN und CPU 317F-2 PN/DP) teilen sich die Submodule eines IO-Devices (ET200S HF). Die beiden IO-Controller befinden sich im gleichen STEP 7 Projekt, dies bietet den Vorteil, dass die Konsistenzprüfung automatisch erfolgt.

Vorgehensweise

Um die Funktion Shared Device nutzen zu können, sind sowohl im SIMOTION SCOUT, SIMATIC Manager als auch in HW Konfig Projektierungsschritte erforderlich.

Vorbereitende Schritte SIMOTION CPU

1. Legen Sie im SIMOTION SCOUT ein Projekt mit dem Namen **Shared-Device-Projekt** an.
2. Fügen Sie eine SIMOTION D445-2 DP/PN ein und projektieren diese.
3. Öffnen Sie die SIMOTION CPU in HW Konfig und projektieren Sie die PROFINET-Schnittstelle.
4. Projektieren Sie ein PROFINET IO-Device ET 200S (IM151-3PN HF) mit einigen Submodulen, wie im Bild dargestellt.

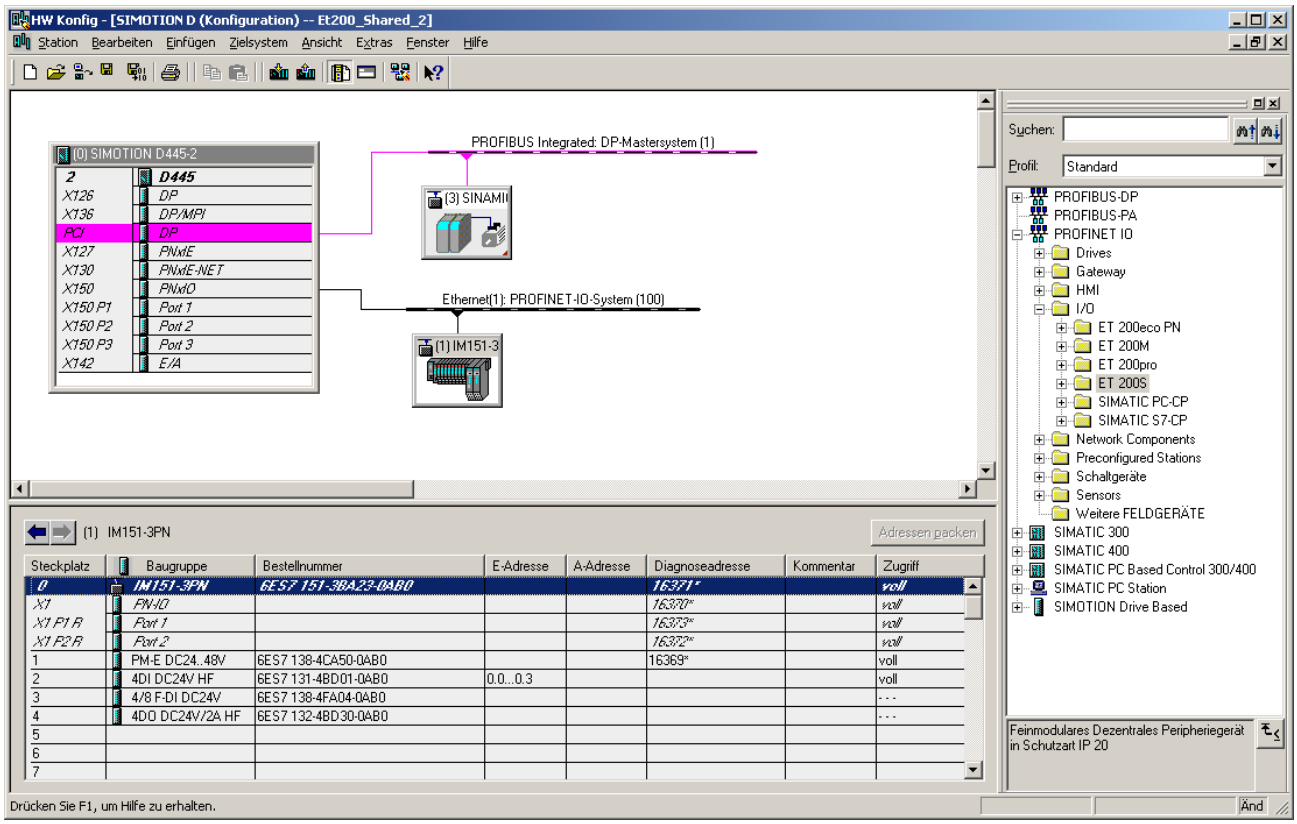


Bild 8-27 SIMOTION D445-2 DP/PN mit IO-Device ET200S

5. **Speichern und Übersetzen** Sie in HW Konfig.

Vorbereitende Schritte SIMATIC CPU

1. Öffnen Sie das erstellte Projekt im SIMATIC Manager.
2. Fügen Sie zusätzlich eine SIMATIC 300 Station und öffnen diese in HW Konfig.
3. Fügen Sie z. B. eine CPU 317F-2 PN/DP ein und projektieren Sie die PROFINET-Schnittstelle.
4. **Speichern und Übersetzen** Sie in HW Konfig.

Shared Device anlegen

1. Öffnen Sie eine der zuvor angelegten SIMOTION CPU in HW Konfig.
2. **Kopieren** Sie das angelegte IO-Device ET200S über das Kontextmenü (rechte Maustaste).
3. **Speichern** Sie die Hardware-Konfiguration und schließen Sie die konfigurierte Station.
4. Öffnen Sie die, zuvor angelegte Station mit der SIMATIC F-CPU in HW Konfig.
5. Um das IO-Device als Shared Device einzufügen, klicken Sie mit der rechten Maustaste auf das PROFINET-IO-System. Wählen Sie im Kontextmenü den Befehl **Shared Einfügen**.

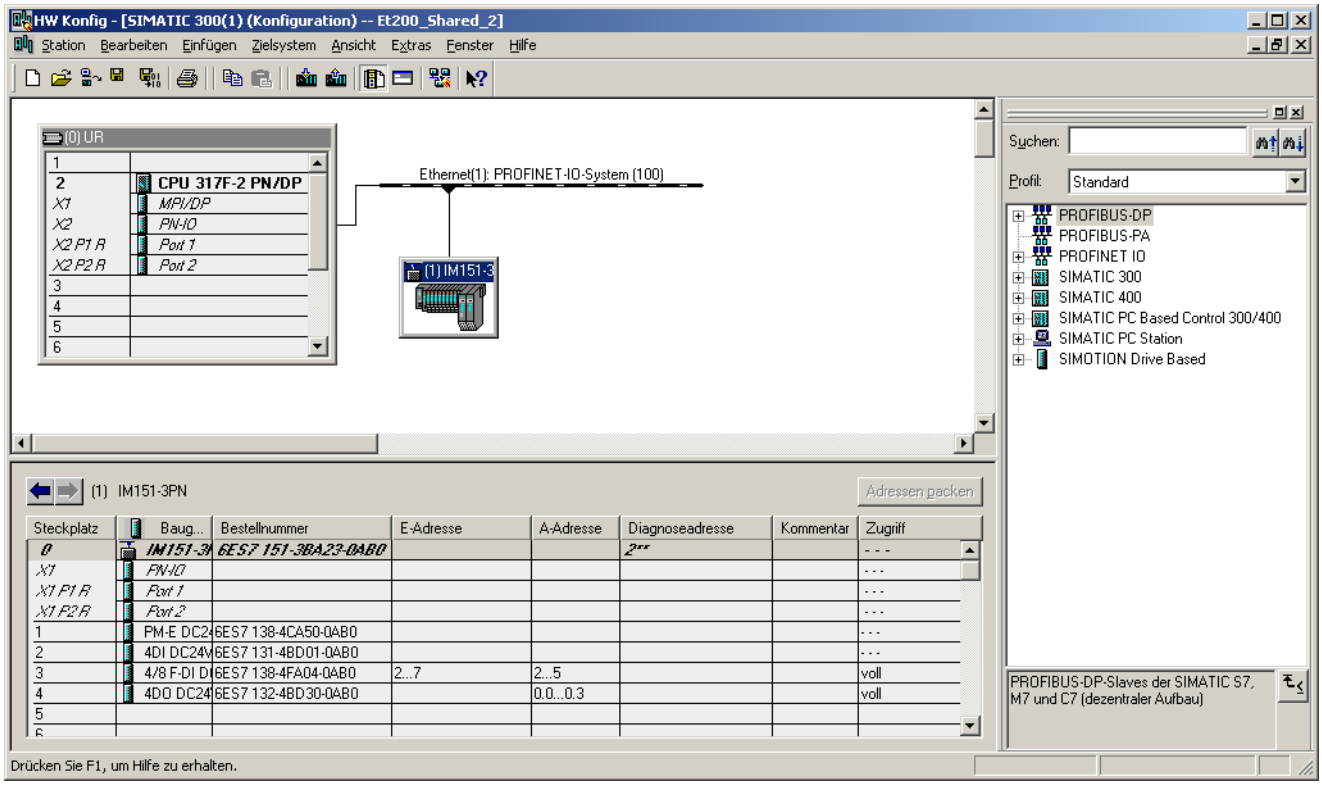


Bild 8-28 SIMATIC CPU 317F-2 PN/DP mit Shared Device ET200S

6. Speichern Sie die Hardware-Konfiguration und schließen Sie die konfigurierte Station.

Sie haben das Shared Device erfolgreich angelegt, parametrieren Sie nun die Zuordnung der Submodule zu den projektierten Stationen.

Submodule zuordnen

Die Zuordnung der Submodule muss für jede Station separat erfolgen. Beachten Sie, dass Änderungen in einer Station Auswirkungen auf die andere(n) Station(en) haben! Ein Submodul kann immer nur einer Station zugeordnet werden!

1. Öffnen Sie den Eigenschaftsdialog des PROFINET IO-Devices der SIMOTION CPU.
2. Klicken Sie auf das Register **Zugriff**.

3. Konfigurieren Sie den Zugriff auf die einzelnen Submodule. Wählen Sie dazu aus der Klappliste in der Spalte **Wert** die Art des Zugriffs aus. Sie können wählen zwischen:
   - Kein Zugriff auf das Submodul: "- - -"
   - Voller Zugriff auf das Submodul: "voll"

   Beachten Sie, dass die Einstellung "voll" in der / den anderen Station(en) automatisch zur Einstellung "- - -" führt.

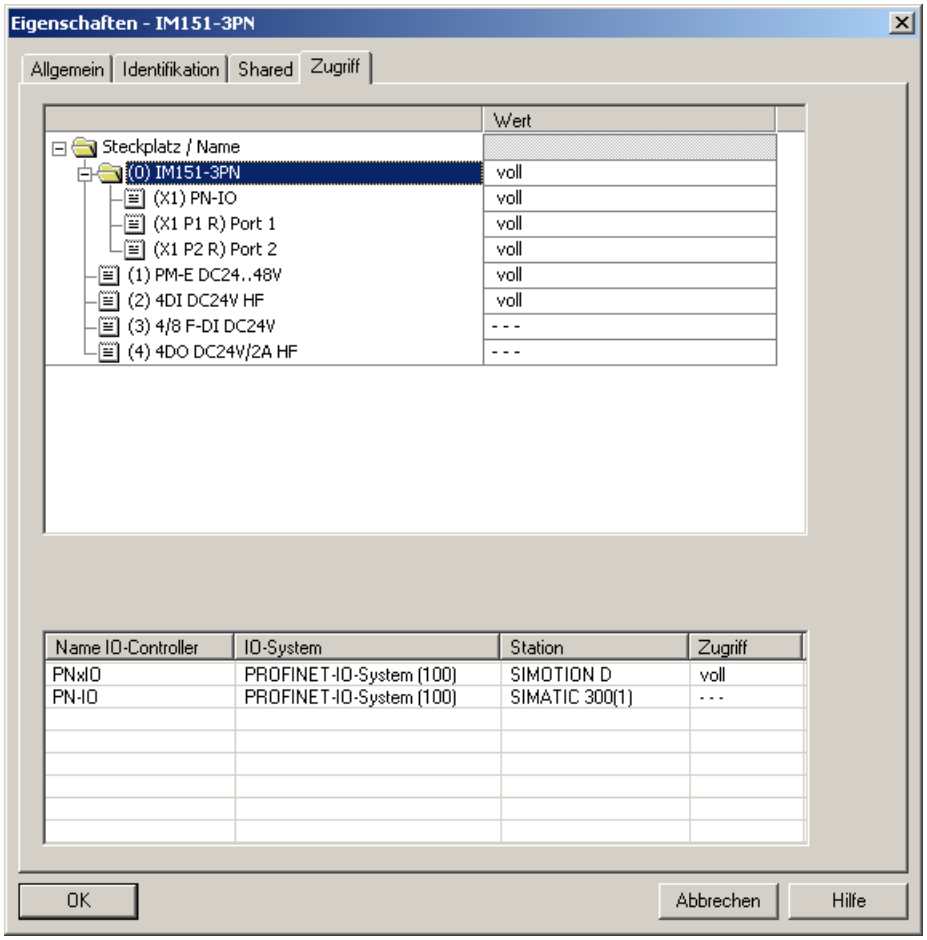


Bild 8-29 Zugriff SIMOTION D445-2 DP/PN auf ET200S

4. Speichern und übersetzen Sie die Station und schließen Sie diese.

5. Wiederholen Sie die Schritte 1 bis 4 für das Shared Device an der SIMATIC F-CPU.

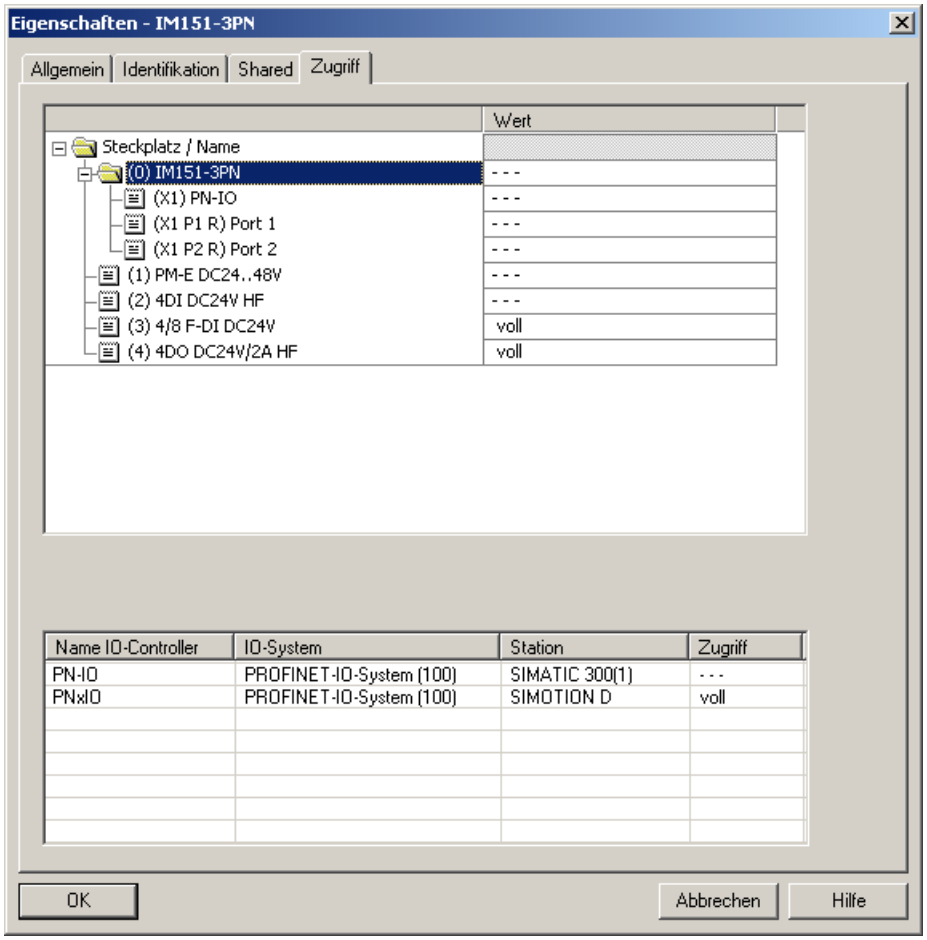


Bild 8-30 Zugriff SIMATIC CPU 317F-2 PN/DP auf Shared Device ET200S

6. Laden Sie abschließend die Konfiguration in die Stationen herunter.

Shared Device im Anwenderprogramm

Dem Shared Device kommt im Anwenderprogramm keine besondere Rolle zu. Die Submodule, die der Station zugeordnet sind, werden wie üblich über ihre Adressen angesprochen, die anderen Submodule erhalten keine Adressen.

8.5.6.3 Projektieren C240 PN und F-CPU mit S120 als Shared Device

Beschreibung

In diesem Kapitel erhalten Sie einen groben Leitfaden zur Projektierung eines SINAMICS S120 als Shared Device. Im Beispiel werden die Motion-Control Aufgaben über die SIMOTION C240 PN durchgeführt und die PROFIsafe Überwachung mit einer SIMATIC F-CPU z. B. 317F-2PN/DP.

Hinweis

Einschränkung

Die Verwendung der Funktion Shared Device mit SINAMICS an SIMOTION hat derzeit noch Einschränkungen. Die Nutzung von Shared Device bei SINAMICS an SIMOTION erfolgt nur, indem bei der Antriebs-Projektierung in HW-Konfig die GSD-Dateien für SINAMICS verwendet werden, kein Device OM. Damit sind Abgleichsmechanismen zwischen SCOUT-Starter-HW-Konfig nicht möglich. Es muss dann in den einzelnen Projektierungsschritten manuell für Konsistenz in der Telegrammprojektierung gesorgt werden.

Daher ist folgende Projektierungsanweisung nur zur weiteren Erläuterung der Funktionalität Shared Device zu sehen. Bei der Nutzung von PROFIsafe auf PROFINET mit SIMOTION und SIANAMICS ist immer die Kommunikation über den I-Device F-Proxy zu verwenden, wie in Kapitel Grundlagen I-Device-F-Proxy (Seite 216) beschrieben.

So projektieren Sie ein Shared Device am Beispiel SINAMCIS S120 und SIMOTION C240 PN

1. Erstellen Sie in SIMOTION SCOUT ein neues Projekt mit einer SIMOTION C240 PN.
2. Öffnen Sie die Hardware in HW Konfig und projektieren Sie die C240 PN als IRT Sync-Master.
3. Installieren Sie die GSD-Datei des S120. Diese befindet sich als ZIP auf der CF-Card des Antriebs im Ordner SIEMENS\SIMATIC\DATA\CFG\ z. B. CBE20GSD.ZIP.
4. Ziehen Sie die installierte GSD per Drag&Drop aus dem Hardwarekatalog **PROFINET IO/SINAMICS/GSD/SINAMICS S120/S150 CU320-2 PN Shared Device** an das PROFINET-Netz.

5. Fügen Sie die Subslots **DO Control Unit** und **DO Servo** aus der GSD im Hardwarekatalog in den Baugruppenträger ein und konfigurieren Sie die Telegramme u. a. **PROFIsafe Telegr. 30**.

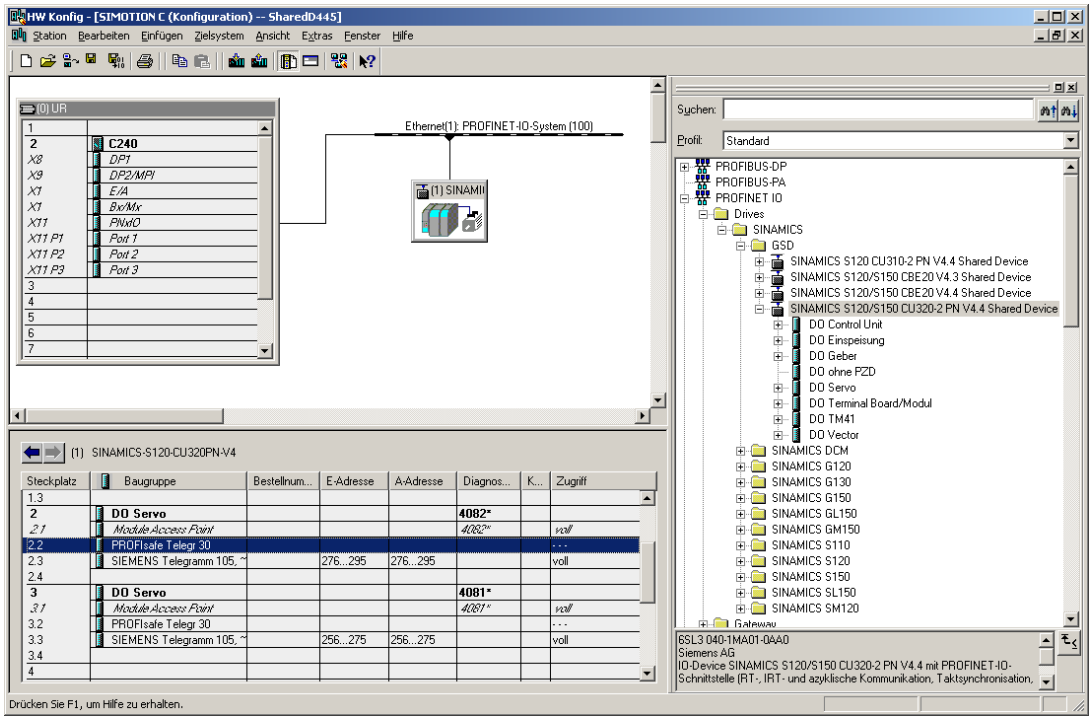


Bild 8-31 Shared Device S120 an SIMOTION C240 PN

6. Öffnen Sie die Objekteigenschaften des Antriebsgeräts und stellen Sie im Register **Zugriff** die Slots ein. Beim Slot **PROFIsafe Telegr. 30** muss als Wert "**- - -** " eingestellt sein.

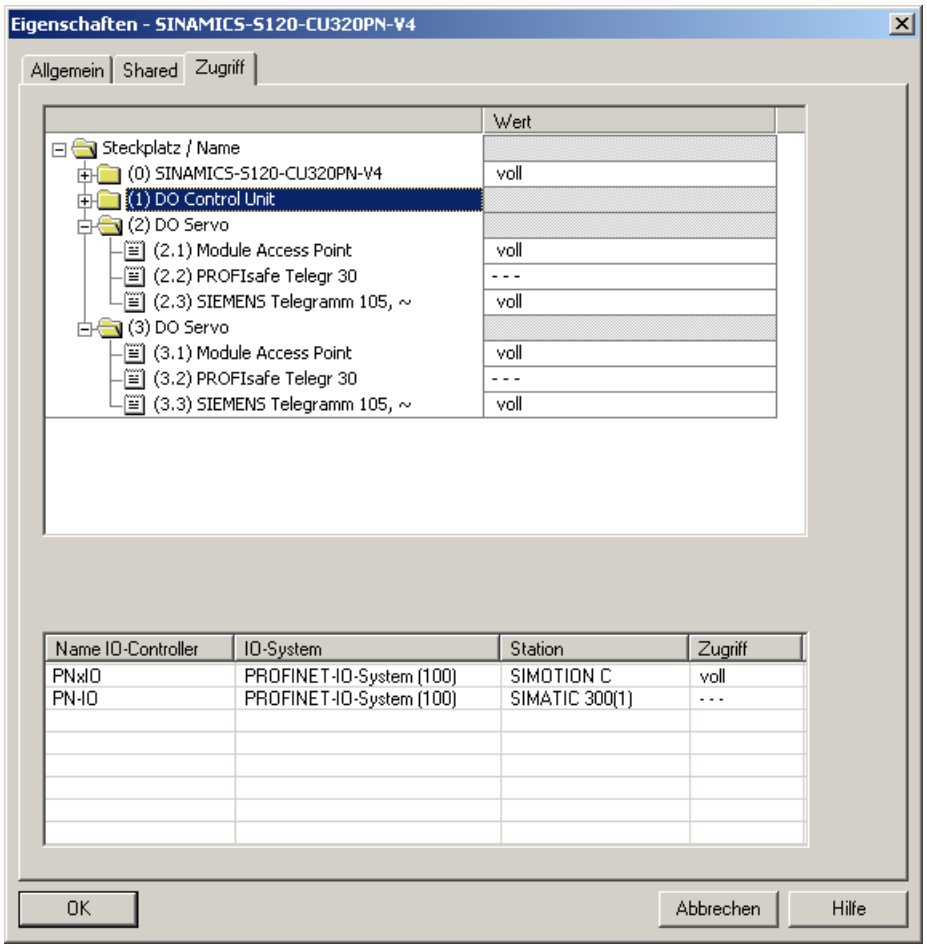


Bild 8-32 Shared Device S120 Zugriff von C240 PN

7. Projektieren Sie den Antrieb als Sync-Slave und taktsynchron zum Servo der C240 PN und verschalten Sie die Topologie.
8. Öffnen Sie das Projekt im SIMATIC Manager und fügen Sie eine SIMATIC 300 Station ein. Öffnen Sie diese in HW Konfig.
9. Fügen Sie in HW Konfig eine SIMATIC CPU 317F-2PN/DP ein. Diese verschalten und projektieren Sie als **nicht synchronisiert**.
10. Wechseln Sie in HW Konfig in das Projekt mit der C240 PN, markieren das Antriebsgerät und wählen **Kopieren** im Kontextmenü.

11. Markieren Sie im Projekt mit der F-CPU das PROFINET-IO-System und wählen Sie im Kontextmenü **Shared Einfügen**. Der S120 wird eingefügt und im Baugruppenträger angezeigt.

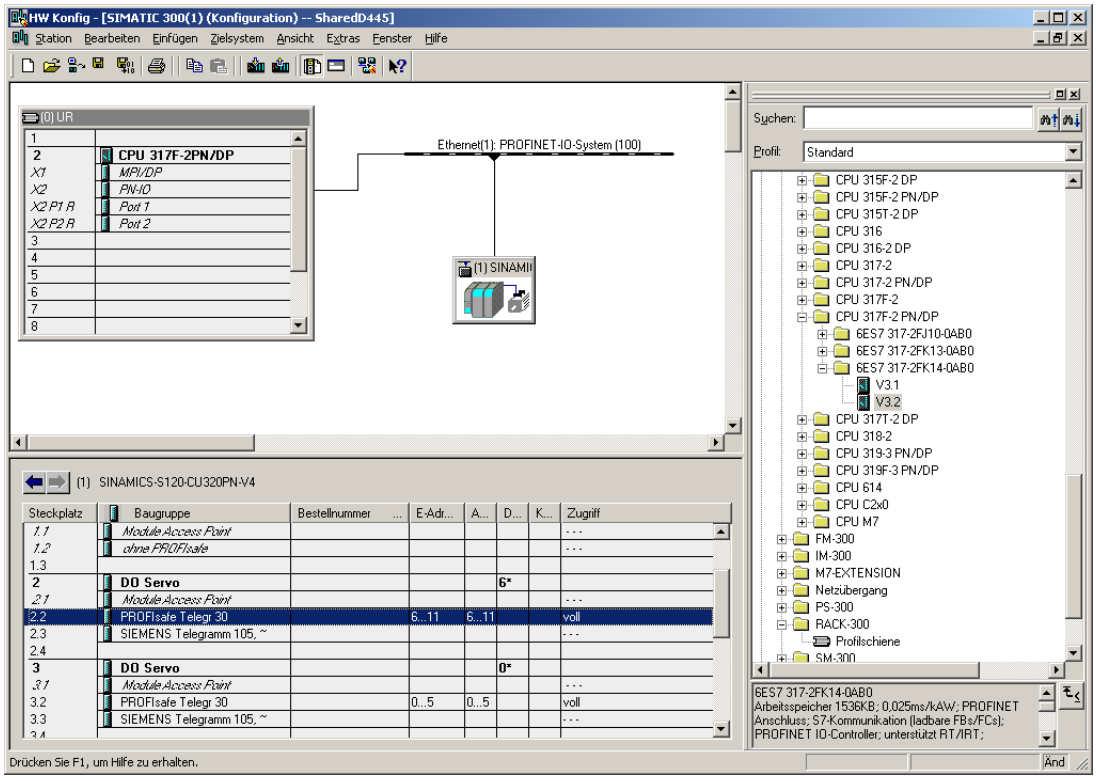


Bild 8-33 Shared Device S120 an SIMATIC CPU 317F-2PN/DP

12. Öffnen Sie die Objekteigenschaften des Antriebsgeräts und stellen Sie im Register **Zugriff** die Slots ein. Beim Slot **PROFIsafe Telegr. 30** muss als Wert "**voll**" eingestellt sein, die anderen bleiben auf "**- - -**".

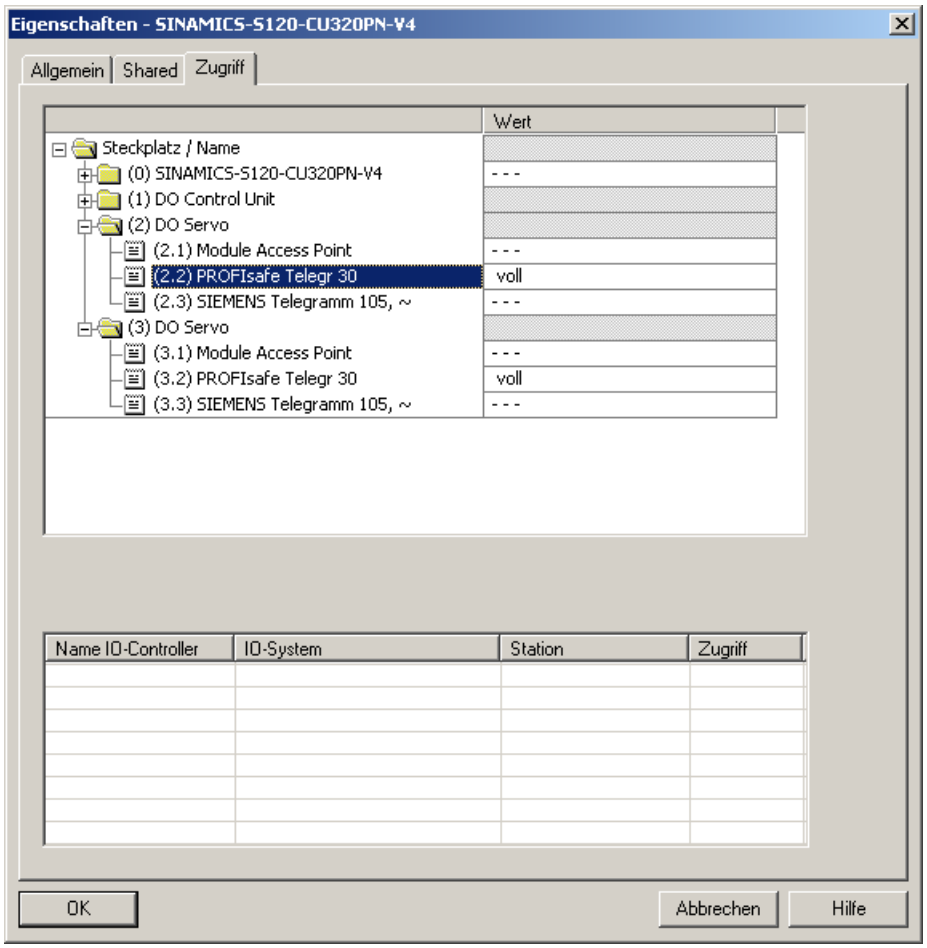


Bild 8-34 Shared Device S120 Zugriff von CPU 317-F

13. Verschalten Sie die Topologie und Speichern und Übersetzen Sie die Projekte.

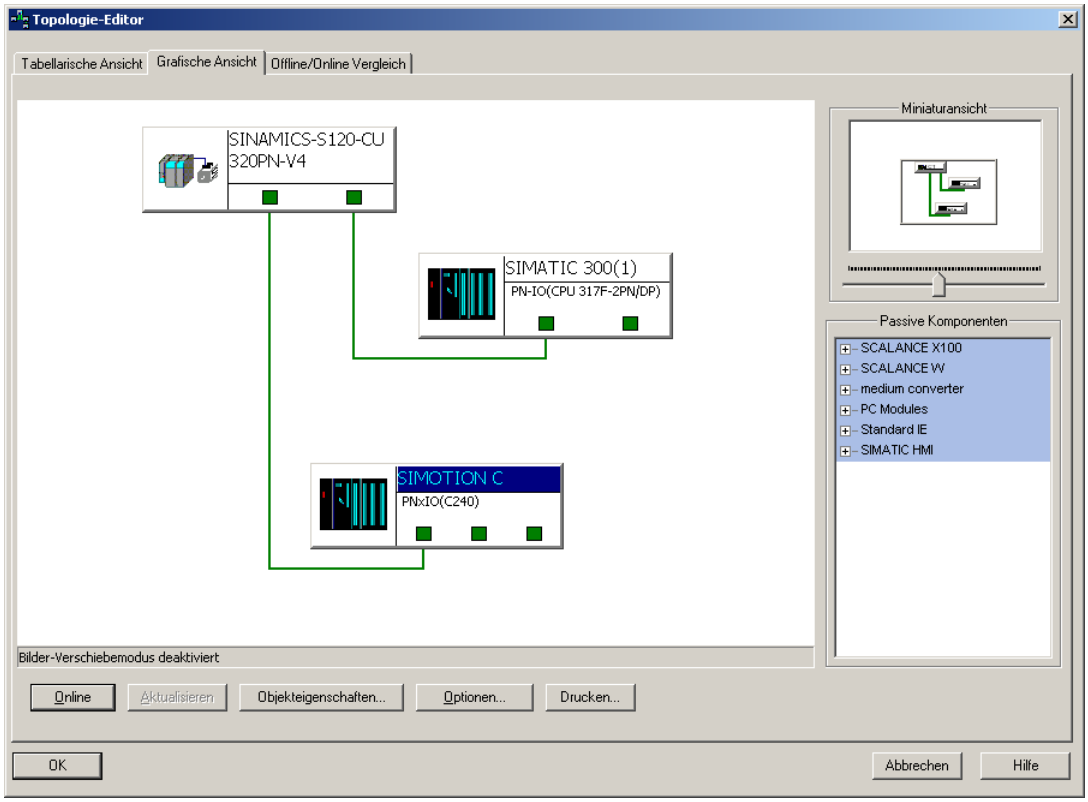


Bild 8-35 Topologie Shared Device S120 an C240 PN und CPU 317-F

14. In SIMOTION SCOUT ist im C240 PN Projekt aufgrund der GSD Integration kein konfigurierbarer Antrieb vorhanden. Fügen Sie ein Einzelantriebsgerät S120 CU320-2 ein und konfigurieren diesen im SIMOTION SCOUT.

Hinweis

Die IP-Adresse und der Gerätename des Einzelantriebsgeräts müssen identisch zum über die GSD eingefügten S120 sein.

---

15. Konfigurieren Sie die Safety-Funktionen und achten Sie bei PROFIsafe auf die F_Dest_Adresse.

16. Die Funktionalität Shared Devices des Antriebs muss über den Parameter p8929 aktiviert werden. Dazu setzen Sie **p8929** auf **(2) Automatisierung und Safety**. Damit haben die zwei PROFINET Controller gleichzeitig Zugriff auf den Antrieb (SIMOTION CPU und SIMATIC F-CPU).
Hinweis:
Bei einer CU320-2 PN über integrierter Schnittstelle: Parameter 8929
Bei einer CU320-2 DP mit CBE20: Parameter 8829

8.6 PROFIsafe über PROFIBUS

8.6.1 Allgemeines zu PROFIsafe an PROFIBUS

Zwei Möglichkeiten für PROFIsafe

Grundsätzlich stehen Ihnen zwei Möglichkeiten zur PROFIsafe Kommunikation am PROFIBUS zur Verfügung:

- I-Slave-F-Proxy

  F-CPU ist als DP-Master im Projekt und überwacht die Antriebe am unterlagerten SIMOTION I-Slave

- F-Querverkehr

  Die SIMOTION ist DP-Master im Projekt und die F-CPU überwacht als I-Slave die Antriebe.

Die Vorgehensweise bei der Projektierung der PROFIsafe-Kommunikation ist weitgehend identisch. In den nachfolgenden Kapiteln finden Sie jeweils ein kurzes Beispiel.

8.6.2 Unterstützte Geräte und Softwarevoraussetzungen PROFIsafe an PROFIBUS

Notwendige Softwarepakete auf dem Programmiergerät:

- SIMATIC Manager STEP7 Version ab 5.4 SP2
- S7 F Configuration Pack Version ab 5.5 SP3
- S7 Distributed Safety Programming Version ab 5.4 SP3
- SIMOTION SCOUT ab Version 4.1.1 HF6
- SINAMICS Firmware-Version ab 2.5

Hinweis

Ab der SIMOTION Firmware 4.1.1 HF10 und SINAMICS Firmware 2.5 SP1 HF10 sind mit einer CX32 5 Antriebe projektierbar. Mit früheren Firmwareversionen sind max. 4 Antriebe projektierbar.

Die für PROFIsafe geeigneten Komponenten finden Sie im *Funktionshandbuch S120 Safety Integrated*.

Unterstützte Geräte

Tabelle 8- 2 Geräteübersicht

SIMOTION CPU	
Controller Based	C240 C240 PN
PC Based	P350-3
Drive Based (blocksize)	D410 DP
Drive Based (booksize)	D425 D435 D445 D445-1 D445-2 DP/PN D455-2 DP/PN
SINAMICS Antriebsgeräte	
S120 CX32	CX32 CX32-2
S120	CU320 DP CU320-2 DP CU310 DP CU310-2 DP
S110	CU305 DP

Unterstützte Anzahl Antriebsachsen

Bei PROFIBUS sind nur 16 Antriebsachsen pro PROFIBUS-Schnittstelle nutzbar.

8.6.3 I-Slave-F-Proxy

8.6.3.1 Grundlagen I-Slave-F-Proxy

Kurzbeschreibung

Mithilfe des I-Slave-F-Proxys können Sie eine PROFIsafe-Projektierung mit F-Host (F-CPU SIMATIC) an PROFIBUS mit SIMOTION Geräten (SIMOTION D, SIMOTION P350, SIMOTION C) für die unterlagerten Antriebe erstellen. Das Routing zyklischer PROFIsafe Daten zu SINAMICS Antriebe am SINAMICS_Integrated bzw. am PROFIBUS DP ist damit möglich.

Ein F-Host kommuniziert über die I-Slave Schnittstelle und einen F-Proxy einer SIMOTION CPU mit den Antrieben, die sich am PROFIBUS DP der SIMOTION CPU befinden können. An den Kommunikationssträngen der SIMOTION CPU befinden sich SINAMICS S120/S110 und SINAMICS integrated/CX32/CX32-2.

8.6.3.2 Topologie I-Slave-F-Proxy für PROFIBUS Antriebsgeräte

Topologiebeispiel I-Slave-F-Proxy

Im folgenden Bild sehen Sie eine schematische Topologiedarstellung, bei der die Safety Antriebe über PROFIBUS DP an die SIMOTION CPU angeschlossen sind und diese über PROFIBUS DP an die F-CPU.

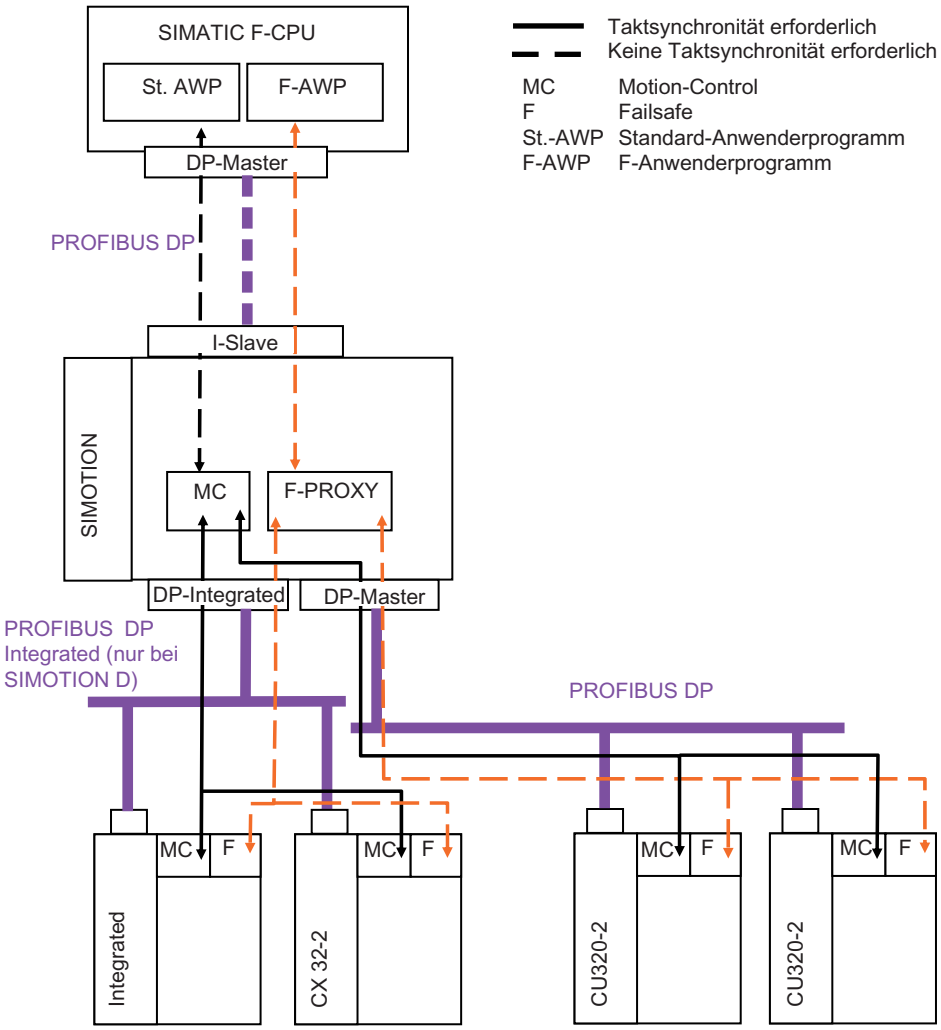


Bild 8-36 Topologie I-Slave-F-Proxy für PROFIBUS Antriebsgeräte

8.6.3.3 PROFIsafe über PROFIBUS bei Verwendung von SIMOTION D

Im Folgenden soll eine PROFIsafe-Kommunikation zwischen dem integrierten Antriebsgerät SINAMICS S120 einer SIMOTION D bzw. CX32 und einer übergeordneten SIMATIC F-CPU über PROFIBUS konfiguriert werden.

Topologieaufbau (Netzsicht der Projektierung)

Die prinzipielle Topologie der an der PROFIsafe-Kommunikation über PROFIBUS beteiligten Komponenten (SIMATIC F-CPU und D4x5 mit SINAMICS S120 integriert bzw. CU320) finden Sie im vorhergehenden Kapitel.

Das Antriebsgerät (SINAMICS) und SIMATIC F-CPU befinden sich an unterschiedlichen PROFIBUS-Subnetzen. In diesem Fall wird ein PROFIsafe-Netzübergang auf SIMOTION D konfiguriert, damit die notwendigen Daten von einen Netz ins andere kopiert werden (siehe dazu auch Punkt 7).

Projektieren der PROFIsafe-Kommunikation

Im Folgenden wird die Projektierung einer PROFIsafe-Kommunikation zwischen einem Antriebsobjekt eines integrierten SINAMICS-Antriebsgeräts einer SIMOTION D und einer SIMATIC F-CPU beschrieben. Das Vorgehen bei der Projektierung einer PROFIsafe-Kommunikation zwischen einem Antriebsgerät einer CU320 und einer SIMATIC F-CPU ist prinzipiell gleich und wird nicht gesondert beschrieben.

1. Legen Sie entsprechend der vorliegenden Hardware in HW Konfig eine F-CPU z. B. CPU 317F-2 und eine SIMOTION D4x5-Steuerung (mit integriertem SINAMICS S120) an.
2. Definieren Sie die gewünschte SIMOTION CPU als DP-Slave und die angeschlossene F-CPU als den dazugehörigen DP-Master.
3. Konfigurieren Sie das SINAMICS-Antriebsgerät mit dem SIMOTION SCOUT entsprechend dem vorliegenden Hardware-Aufbau.
4. Fügen Sie daher ein neues TO Achse ein und durchlaufen Sie den Achsassistenten. Im Assistent verschalten Sie die Achse auf das entsprechende Antriebsobjekt des S120 und automatisch wird ein entsprechendes Telegramm angelegt (Symbolische Zuordnung).
5. Speichern und übersetzen Sie das Projekt.

6. Legen Sie in der Konfiguration des SINAMICS-Antriebsgerätes einen PROFIsafe-Slot an. Selektieren Sie dazu im Register **IF1: PROFIdrive PZD-Telegramme** das Antriebsobjekt, welches über PROFIsafe mit der SIMATIC F-CPU kommunizieren soll. Klicken Sie auf die Schaltfläche **Telegrammkonfiguration anpassen** und wählen Sie **PROFIsafe hinzufügen** aus.

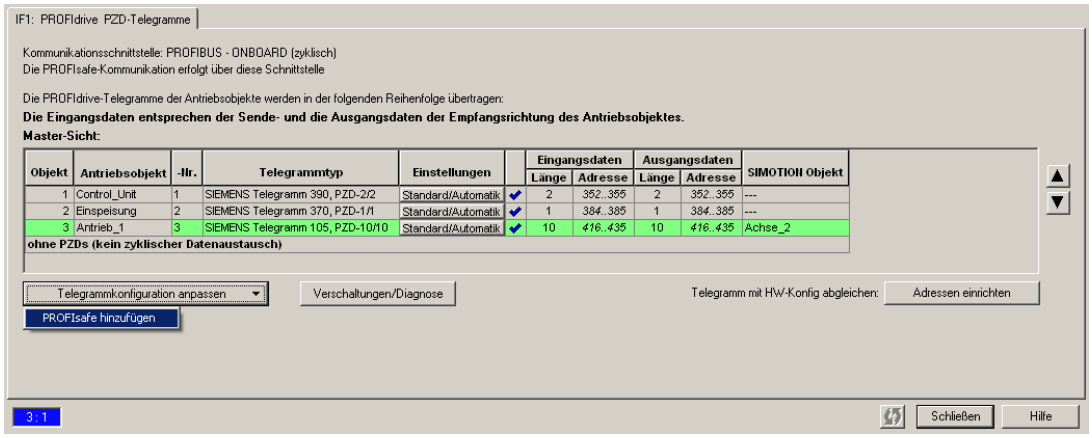


Bild 8-37 PROFIsafe Slot einfügen

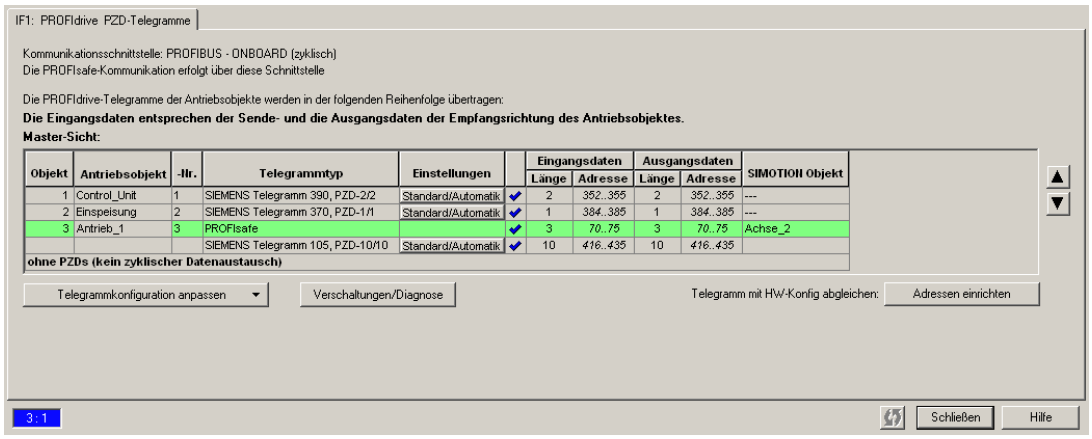


Bild 8-38 PROFIBUS-Telegramm

7. Speichern und übersetzen Sie das Projekt.
8. Anschließend übertragen Sie den neuen PROFIsafe-Slot in HW Konfig durch Klicken auf die Schaltfläche **Adresse einrichten**.

9. Koppeln Sie in HW Konfig der F-CPU die bereits projektierte SIMOTION-Station an die F-CPU (Hardwarekatalog: **PROFIBUS DP > bereits projektierte Stationen** ...).

Hinweis

Für die Konfiguration der SINAMICS Safety Integrated Extended Functions durch SIMOTION ist zusätzlich eine Telegrammerweiterung erforderlich. Ein Safety Datenblock wird als Erweiterung an das PROFIdrive-Istwerttelegramm angehängt. Konfiguration und Parametrierung dieses Safety Datenblocks sind im Funktionshandbuch *SIMOTION Motion Control TO Achse elektrisch/hydraulisch, Externer Geber* beschrieben.

10. Über die Eigenschaften des DP-Slaves (Doppelklick auf SIMOTION I-Slave) in der Lasche **F-Konfiguration** werden nach klicken auf **Neu** die Parameter für die F-Kommunikation angezeigt.

Mode: Zeigt die Kommunikationsbeziehung an. F-MS-Module steht für eine sicherheitsgerichtete Master-Slave-Kommunikation mit SIMOTION.

DP-Partner (F-Peripherie): Eigenschaften des SINAMICS-Antriebs.
Hier kann über **DP Adresse** oder **Adresse** der entsprechende PROFIsafe Antrieb ausgewählt werden.

lokal: Eigenschaften der SIMOTION-CPU.
Hier muss unter "Adresse" die logische Anfangsadresse für die F-Kommunikation seitens der SIMOTION-CPU eingetragen werden.
Der Adressbereich für das Senden und Empfangen der Sicherheitstelegramme ist 6 Byte und muss außerhalb des Prozessabbildes der SIMOTION-CPU (>=64) liegen.

Master (Sicherheitsprogramm): Eigenschaften der SIMATIC F-CPU.
Hier muss unter "Adresse" (LADDR) die logische Anfangsadresse für die F-Kommunikation seitens der SIMATIC F-CPU eingetragen werden. Der Adressbereich für das Senden und Empfangen der Sicherheitstelegramme ist 6 Byte und muss innerhalb des Prozessabbildes der SIMATIC F-CPU liegen.
Im Sicherheitsprogramm der SIMATIC F-CPU kann über diesen Adressbereich auf das PROFIsafe-STW bzw. -ZSW zugegriffen werden.

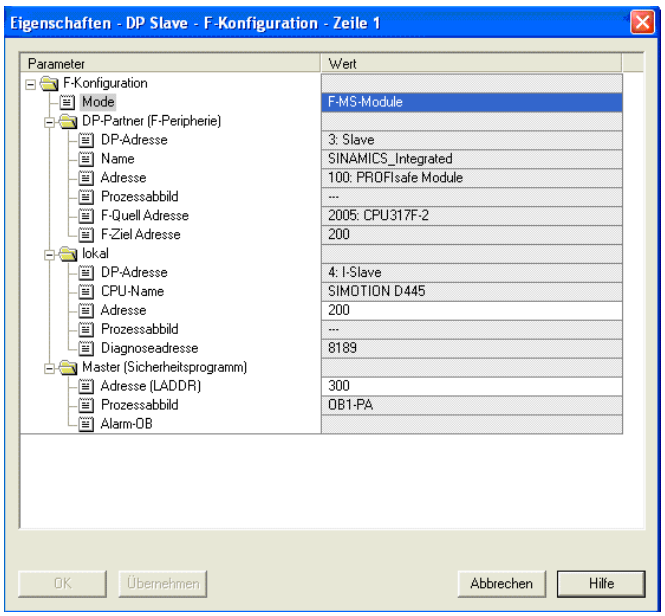


Bild 8-39 Master-Slave-Kopplung in PROFIsafe

11. Öffnen Sie in HW Konfig das Projekt der SIMOTION-CPU.

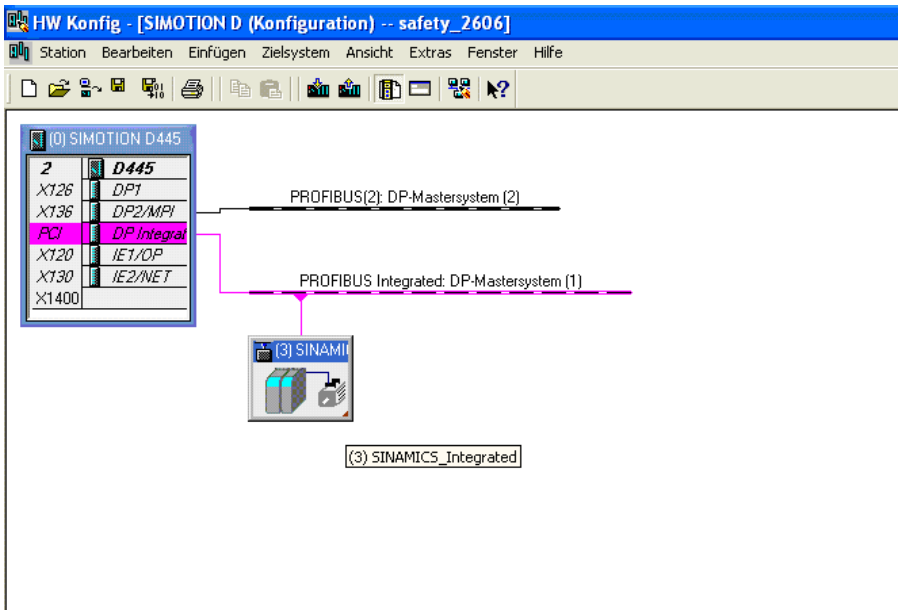


Bild 8-40 SIMOTION D Konfiguration

12. Doppelklicken Sie auf das Symbol des SINAMICS-Antriebsgeräts und wählen Sie im Register **Konfiguration** das Register **Details** aus.

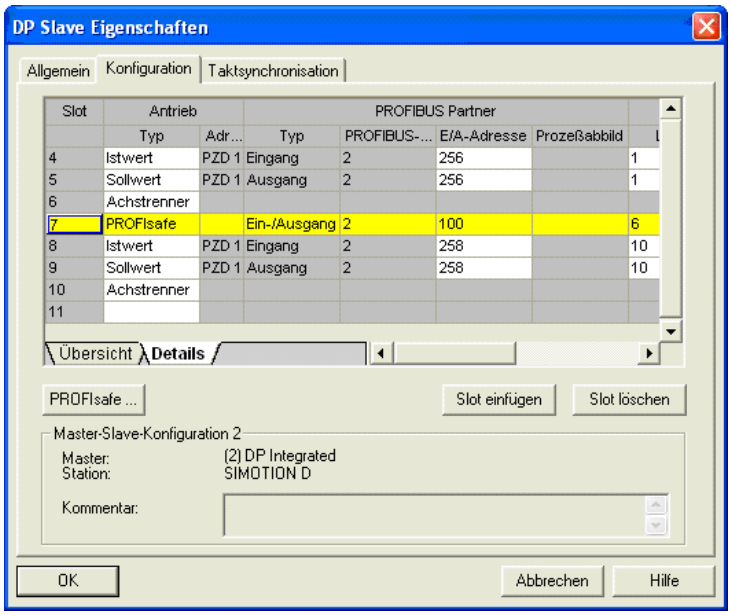


Bild 8-41 Konfiguration von PROFIsafe für SINAMICS-Antriebsgerät

13. Über den Button **PROFIsafe…** legen Sie die für die F-Kommunikation relevanten F-Parameter fest. Ab Step7 V5.5 wird standardmäßig PROFIsafe V2 verwendet.
(Falls der Button **PROFIsafe...** nicht bedienbar ist, müssen Sie den Button über die Schaltfläche **Aktivieren...** Freischalten.)

    Weitere Informationen zu den F-Parametern finden Sie unter Eigenschaften PROFIsafe bei der Projektierung (Seite 214)

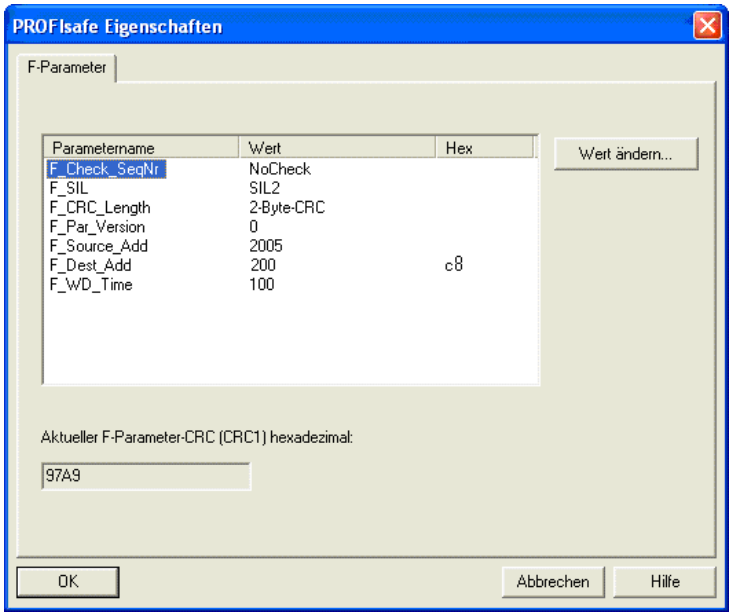


Bild 8-42 PROFIsafe Eigenschaften (F-Parameter)

14. Übersetzen Sie die HW Konfig der SIMOTION-CPU. Anschließend übersetzen Sie die Projektierungen der F-CPU in HW Konfig.

Hinweis

Informationen zur Erstellung eines Sicherheitsprogramms und den Zugriff im Sicherheitsprogramm auf PROFIsafe-Nutzdaten (z. B. STW und ZSW) finden Sie im Programmier- und Bedienhandbuch *SIMATIC, S7 Distributed Safety - Projektieren und Programmieren*.

Safety-Projektierung (online) im SINAMICS-Antrieb

1. Über den Strukturbaum-Eintrag "Funktionen" beim SINAMICS-Antrieb rufen Sie die Konfiguration für Safety Integrated auf.
2. Konfigurieren Sie Safety Integrated und stellen Sie unter der **PROFIsafe-Adresse** des Antriebs (p9610/p9810) den bereits definierten Parameter F_Dest_Add in Hex-Darstellung ein.
3. Führen Sie abschließend ein POWER ON durch. Dadurch wird die Safety-Projektierung im Antrieb wirksam.

Hinweis

Weitere Informationen zur Safety Projektierung finden Sie im Funktionshandbuch SINAMICS S120 Safety Integrated.

8.6.4 F-Querverkehr

8.6.4.1 Grundlagen F-Querverkehr

Funktionsweise

SIMOTION CPU ist DP Master für einen F-Querverkehr. Die SIMATIC F-CPU ist DP Slave am PROFIBUS DP und leitet die F-Kommunikation z. B. mit einer CU320 des SINAMICS S120.

Hinweis

Ein durchrouten der Ansteuerung der Safety-Integrated-Funktionen auf den SINAMICS Integrated der SIMOTION D, Controller Extension CX32/CX32-2 oder eines anderen DP-Netzes ist in dieser Konstellation nicht möglich.

8.6.4.2 Topologie F-Querverkehr über PROFIBUS

Topologiebeispiel F-Querverkehr PROFIBUS

Im folgenden Bild sehen Sie eine schematische Topologiedarstellung, bei der die Safety Antriebe über PROFIBUS DP an die SIMOTION CPU angeschlossen sind und die SIMATIC F-CPU als PROFIBUS I-Slave für die F-Kommunikation.

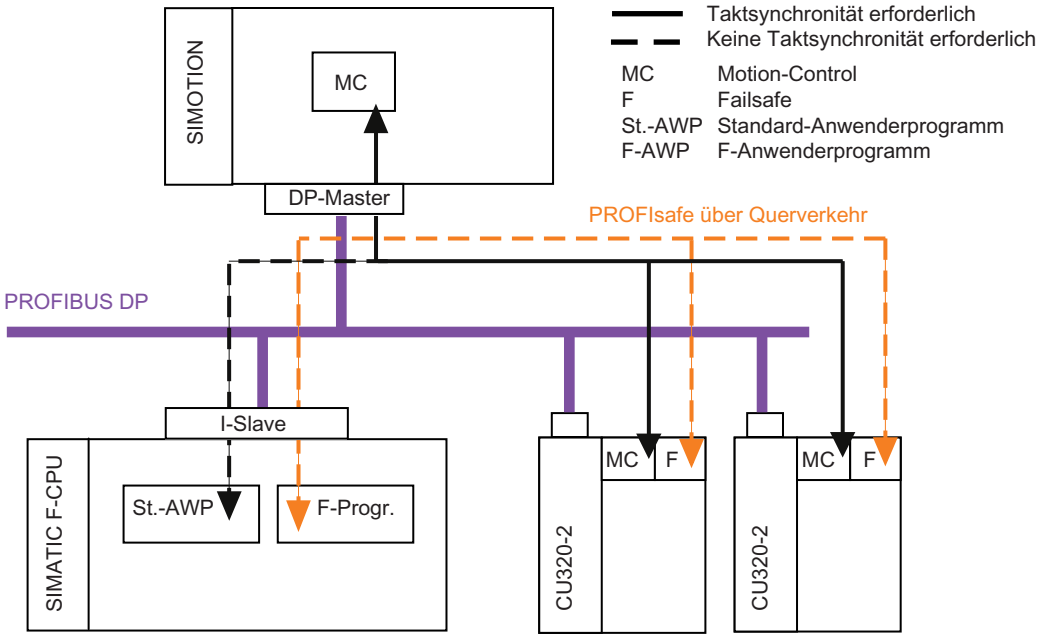


Bild 8-43 Topologie F-Querverkehr für PROFIBUS Antriebsgeräte

8.6.4.3 PROFIsafe über PROFIBUS mit F-Querverkehr am Bespiel SIMOTION D

Im Folgenden soll eine PROFIsafe-Kommunikation zwischen einem SINAMICS S120, einer SIMOTION D als DP Master und SIMATIC F-CPU als I-Slave über PROFIBUS konfiguriert werden.

Topologieaufbau (Netzsicht der Projektierung)

Die prinzipielle Topologie der an der PROFIsafe-Kommunikation über PROFIBUS beteiligten Komponenten (SIMATIC F-CPU und D4x5 mit SINAMICS S120 integriert bzw. CX32) finden Sie im vorhergehenden Kapitel.

Das Antriebsgerät (SINAMICS) und SIMATIC F-CPU befinden sich im gleichen PROFIBUS-Subnetz.

Projektieren der PROFIsafe-Kommunikation über F-Querverkehr

1. Legen Sie entsprechend der vorliegenden Hardware in HW Konfig eine F-CPU z. B. CPU 317F-2, eine SIMOTION D4x5-Steuerung und einen SINAMICS S120 CU320 an.
2. Definieren Sie die gewünschte SIMOTION CPU als DP-Master und die angeschlossene F-CPU als den dazugehörigen DP-Slave.
3. Konfigurieren Sie das SINAMICS-Antriebsgerät mit dem SIMOTION SCOUT entsprechend dem vorliegenden Hardware-Aufbau.
4. Fügen Sie daher ein neues TO Achse ein und durchlaufen Sie den Achsassistenten. Im Assistent verschalten Sie die Achse auf das entsprechende Antriebsobjekt des S120 und automatisch wird ein entsprechendes Telegramm angelegt (Symbolische Zuordnung).
5. Speichern und übersetzen Sie das Projekt.
6. Legen Sie in der Konfiguration des SINAMICS-Antriebsgerätes einen PROFIsafe-Slot an. Selektieren Sie dazu im Register IF1: PROFIdrive PZD-Telegramme das Antriebsobjekt, welches über PROFIsafe mit der SIMATIC F-CPU kommunizieren soll. Klicken Sie auf die Schaltfläche **Telegrammkonfiguration anpassen** und wählen Sie **PROFIsafe hinzufügen** aus.

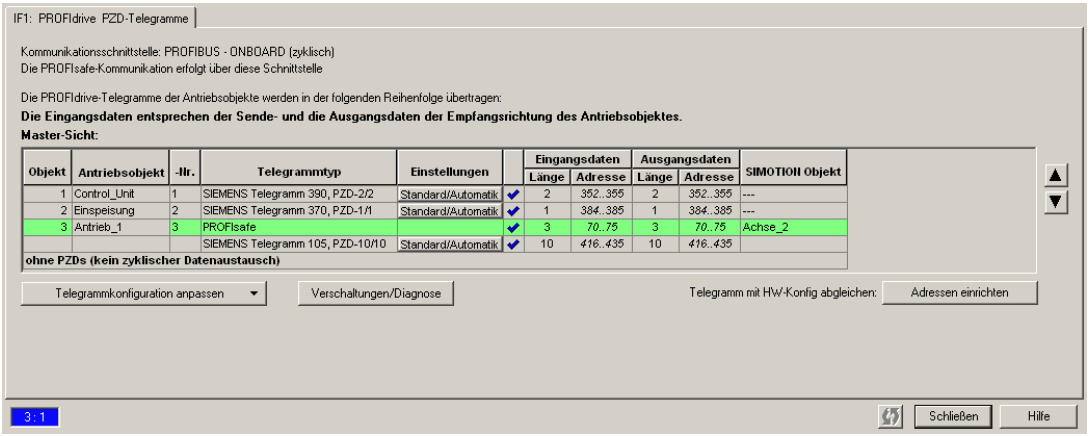


Bild 8-44 PROFIBUS-Telegramm

7. Speichern und übersetzen Sie das Projekt.
8. Anschließend übertragen Sie den neuen PROFIsafe-Slot in HW-Konfig durch Klicken auf die Schaltfläche **Adresse einrichten**.
9. Koppeln Sie in HW Konfig der SIMOTION-Station die bereits projektierte F-CPU an die SIMOTION-Station (HW Katalog: **PROFIBUS DP > bereits projektierte Stationen...).**

10. Über die Eigenschaften des DP-Slaves (F-CPU) in der Lasche **F-Konfiguration** werden die Parameter für die F-Kommunikation angezeigt.

    Mode: Zeigt die Kommunikationsbeziehung an. Für Querverkehr müssen F-DX-Module gewählt werden. Dies steht für eine sicherheitsgerichtete I-Slave-Slave Beziehung.

    DP-Partner (F-Peripherie): Eigenschaften des SINAMICS-Antriebs.
    Hier kann über DP Adresse oder Adresse der entsprechende PROFIsafe Antrieb ausgewählt werden.

    lokal (Sicherheitsprogramm): Eigenschaften der SIMATIC F-CPU.
    Hier muss unter **Adresse (LADDR)** die logische Anfangsadresse für die F-Kommunikation seitens der SIMATIC F-CPU eingetragen werden. Der Adressbereich für das Senden und Empfangen der Sicherheitstelegramme ist 6 Byte und muss innerhalb des Prozessabbildes der SIMATIC F-CPU liegen.
    Im Sicherheitsprogramm der SIMATIC F-CPU kann über diesen Adressbereich auf das PROFIsafe-STW bzw. -ZSW zugegriffen werden.

    Master-Adresse: Eigenschaften der SIMOTION-CPU.
    Hier muss unter **Eingangs-Adresse** die logische Anfangsadresse für die F-Kommunikation seitens der SIMOTION-CPU eingetragen werden.
    Der Adressbereich für das Senden und Empfangen der Sicherheitstelegramme ist 6 Byte und muss außerhalb des Prozessabbildes der SIMOTION-CPU (>=64) liegen.

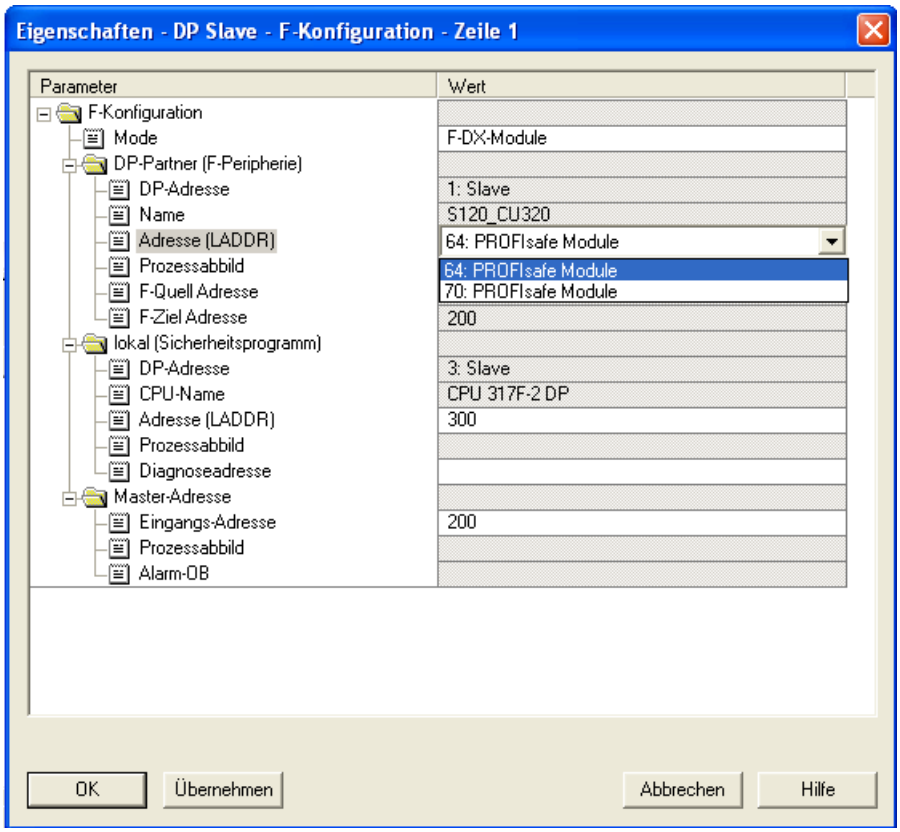


Bild 8-45 Eigenschaften F-Konfiguration Querverkehr

11. Öffnen Sie in HW Konfig das Projekt der SIMOTION-CPU.

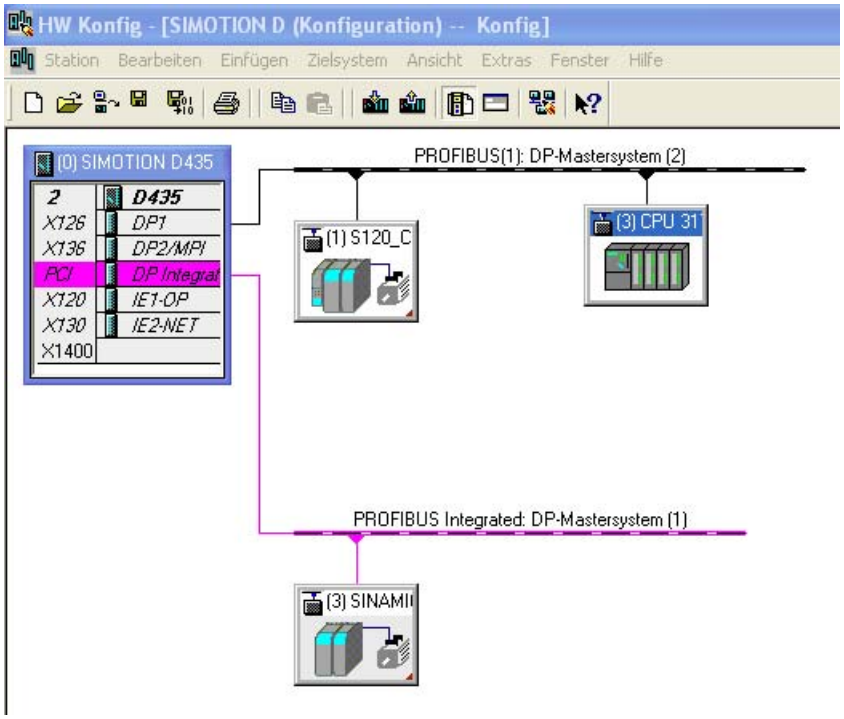


Bild 8-46 HW Konfig F-Konfiguration Querverkehr

12. Doppelklicken Sie auf das Symbol des SINAMICS-Antriebsgeräts und wählen Sie im Register **Konfiguration** das Register **Details** aus. Über den Button **PROFIsafe…** legen Sie die für die F-Kommunikation relevanten F-Parameter fest.
(Falls der Button **PROFIsafe…** nicht bedienbar ist, müssen Sie den Button über die Schaltfläche **Aktivieren...** freischalten.)

    Weitere Informationen zu den F-Parametern finden Sie unter Eigenschaften PROFIsafe bei der Projektierung (Seite 214) .

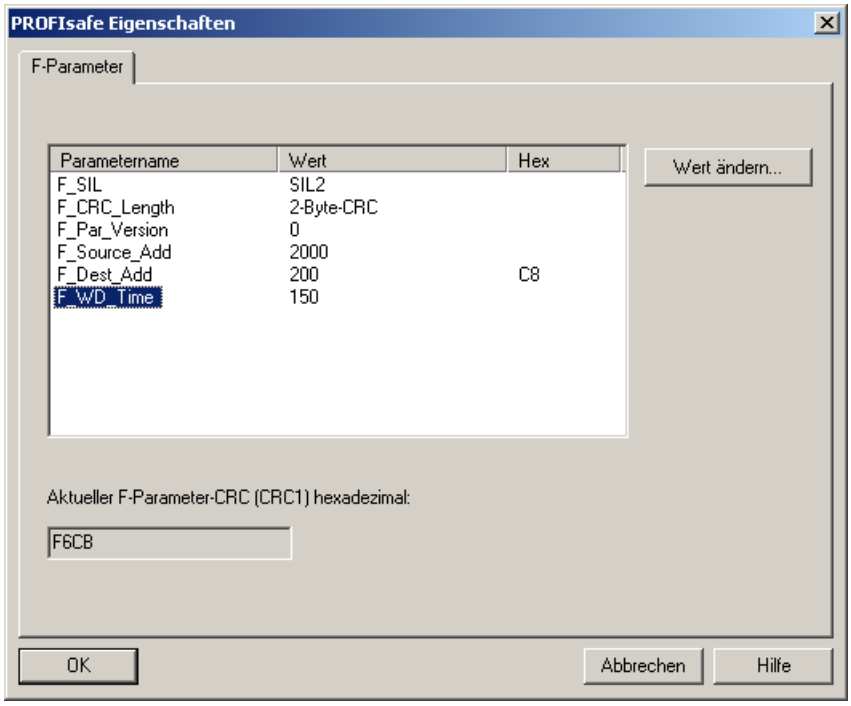


Bild 8-47 PROFIsafe-Eigenschaften F-Querverkehr Mode V1

13. Übersetzen Sie die HW Konfig der SIMOTION-CPU. Anschließend übersetzen Sie die Projektierungen der F-CPU in HW Konfig.

Safety-Projektierung (online) im SINAMICS-Antrieb

1. Über den Strukturbaum-Eintrag **Funktionen** beim SINAMICS-Antrieb rufen Sie die Konfiguration für Safety Integrated auf.
2. Konfigurieren Sie Safety Integrated und stellen Sie unter der PROFIsafe-Adresse des Antriebs (p9610/p9810) den bereits definierten Parameter F_Dest_Add in Hex-Darstellung ein.
3. Führen Sie abschließend ein POWER ON durch. Dadurch wird die Safety-Projektierung im Antrieb wirksam.

Hinweis

Weitere Informationen zur Safety Projektierung finden Sie im Funktionshandbuch *SINAMICS S120 Safety Integrated*.

8.7 PROFIsafe Projektierung - Abnahmetest und -protokolle

Abnahmetests und Abnahmeprotokolle

Nach dem Abschluss der Projektierung und Inbetriebnahme ist für die Sicherheitsfunktionen im Antrieb ein Abnahmetest durchzuführen. Dabei wird die korrekte Parametrierung der Sicherheitsfunktionen überprüft. Die durchgeführten Tests werden in Protokollen dokumentiert.

Hinweis

Für die Abnahme der PROFIsafe-Kommunikation beachten Sie bitte die Informationen im Programmier- und Bedienhandbuch *SIMATIC, S7 Distributed Safety - Projektieren und Programmieren* und das Funktionshandbuch *SINAMICS S120 Safety Integrated* des verwendeten Antriebs.

8.8 Weitere Informationen zu SIMOTION und PROFIsafe

Beschreibung

Weitere Informationen zum Thema PROFIsafe erhalten Sie in folgenden Dokumentationen:

- Wie Sie eine Achse mit einem SINAMICS-Antrieb mit Safety Integrated verschalten, finden Sie beschrieben im Funktionshandbuch TO Achse / Externer Geber.
- Wie Sie einen SINAMICS S120 Antrieb bzw. einen SINAMICS S110 Antrieb mit Safety Integrated projektieren, finden Sie beschrieben in den Handbüchern
  - *Funktionshandbuch SINAMICS S120*
  - *Funktionshandbuch SINAMICS S120 Safety Integrated*
  - *Funktionshandbuch SINAMICS S110*.

PROFIdrive

9

9.1 Warum Profile

Profile in der Automatisierungstechnik legen für Geräte, Gerätefamilien oder gesamte Systeme bestimmte Eigenschaften und Verhaltensweisen so fest, dass dadurch deren weit gehende, eindeutige Charakterisierung erreicht wird. Nur Geräte mit herstellerübergreifend gleichem Profil können sich an einem Feldbus "interoperabel" verhalten und damit die Vorteile eines Feldbusses für den Anwender voll erschließen.

Profile sind von Herstellern und Anwendern getroffene Festlegungen (Spezifikationen) über bestimmte Eigenschaften, Leistungsmerkmale und Verhaltensweisen von Geräten und Systemen. Sie haben das Ziel, Geräte und Systeme, die auf Grund einer "profilgemäßen" Entwicklung zu einer Produktfamilie gehören, an einem Bus interoperabel und bis zu einem gewissen Grad austauschbar betreiben zu können.

Man unterscheidet bei den Profilen zwischen so genannten Applikationsprofilen (allgemeinen oder spezifischen) und Systemprofilen.

- Applikationsprofile beziehen sich vorrangig auf Geräte, in diesem Fall Antriebe, und enthalten sowohl eine vereinbarte Auswahl an Buskommunikation als auch an spezifischen Geräteanwendungen.
- Systemprofile beschreiben Klassen von Systemen unter Einschluss der Masterfunktionalität, Programminterfaces und Integrationsmittel.

PROFIdrive

Das Profil PROFIdrive gehört zu den spezifischen Applikationsprofilen. Es beschreibt im Detail die sinnvolle Anwendung der Kommunikationsfunktionen Querverkehr, Äquidistanz und Taktsynchronisierung in Antriebsapplikationen. Ferner werden alle Geräteeigenschaften, die Einfluss auf die Schnittstelle zu einem über PROFIBUS oder PROFINET verbundenen Controller haben, klar spezifiziert. Dazu gehören u. a. die State maschine (Ablaufsteuerung), das Geberinterface, die Normierung von Werten, die Definition von Standardtelegrammen, der Zugriff auf Antriebsparameter, die Antriebsdiagnose usw.

Das Profil PROFIdrive unterstützt dabei sowohl zentrale als auch dezentrale Motion Control Konzepte.

Die Grundphilosophie: – Keep it simple –

Das Profil PROFIdrive verfolgt die Grundphilosophie, dass die Antriebsschnittstelle so einfach wie möglich und frei von technologischen Funktionen gehalten wird. Durch diese Philosophie haben Referenzmodelle wie auch die Funktionalität und Performance des PROFIBUS-/PROFIRIVE-Masters keinen bzw. nur geringen Einfluss auf die Antriebsschnittstelle.

9.2 PROFIdrive Übersicht

Das PROFIdrive Profil

Das PROFIdrive Profil definiert das Geräteverhalten und das Zugriffsverfahren auf Antriebsdaten für elektrische Antriebe am PROFIBUS und am PROFINET, vom einfachen Frequenzumrichter bis hin zu hochperformanten Servoreglern.

PROFIdrive ist in einen allgemeinen Teil und einen busspezifischen Teil gegliedert.´Im allgemeinen Teil sind folgende Eigenschaften definiert:

- Basismodell (Base Model)
- Parametermodell (Parameter Model)
- Applikationsmodell (Application Model)

Im busspezifischen Teil finden die folgenden Zuordnungen statt:

- PROFIdrive auf PROFIBUS
- PROFIdrive auf PROFINET

Eine genaue Beschreibung des PROFIdrive Profils finden Sie unter nachfolgendem Hinweis.

Literatur-Hinweis

PROFIdrive Profil

PROFIBUS Profile PROFIdrive – Profile Drive Technology
Version V4.1, May 2006,
PROFIBUS User Organization e. V.
Haid-und-Neu-Straße 7, D-76131 Karlsruhe
http://www.profibus.com
Order Number 3.172, spez. Kap. 6

Normen

Norm IEC 61800

9.3 PROFIdrive Basis/Parameter Modell

Beschreibung

Das PROFIdrive Basismodell beschreibt ein Automatisierungssystem als eine Menge von Geräten sowie deren Beziehungen untereinander (Applikationsschnittstellen, Parameterzugriff). Folgende Geräteklassen werden im Basismodell unterschieden:

PROFIdrive	PROFIBUS DP	PROFINET IO
Controller (übergeordnete Steuerung oder Host des Automatisierungssystems)	DP Master Klasse 1	IO Controller
Peripheral Device (P-Device)	DP Slave (I-Slaves)	IO Device
Supervisor (Engineering Station)	DP Master Klasse 2	IO Supervisor

PROFIdrive Geräteklassen

Beispiel für ein PROFIdrive Automatisierungskonzept

Die nachfolgende Grafik zeigt ein typisches Automatisierungskonzept.

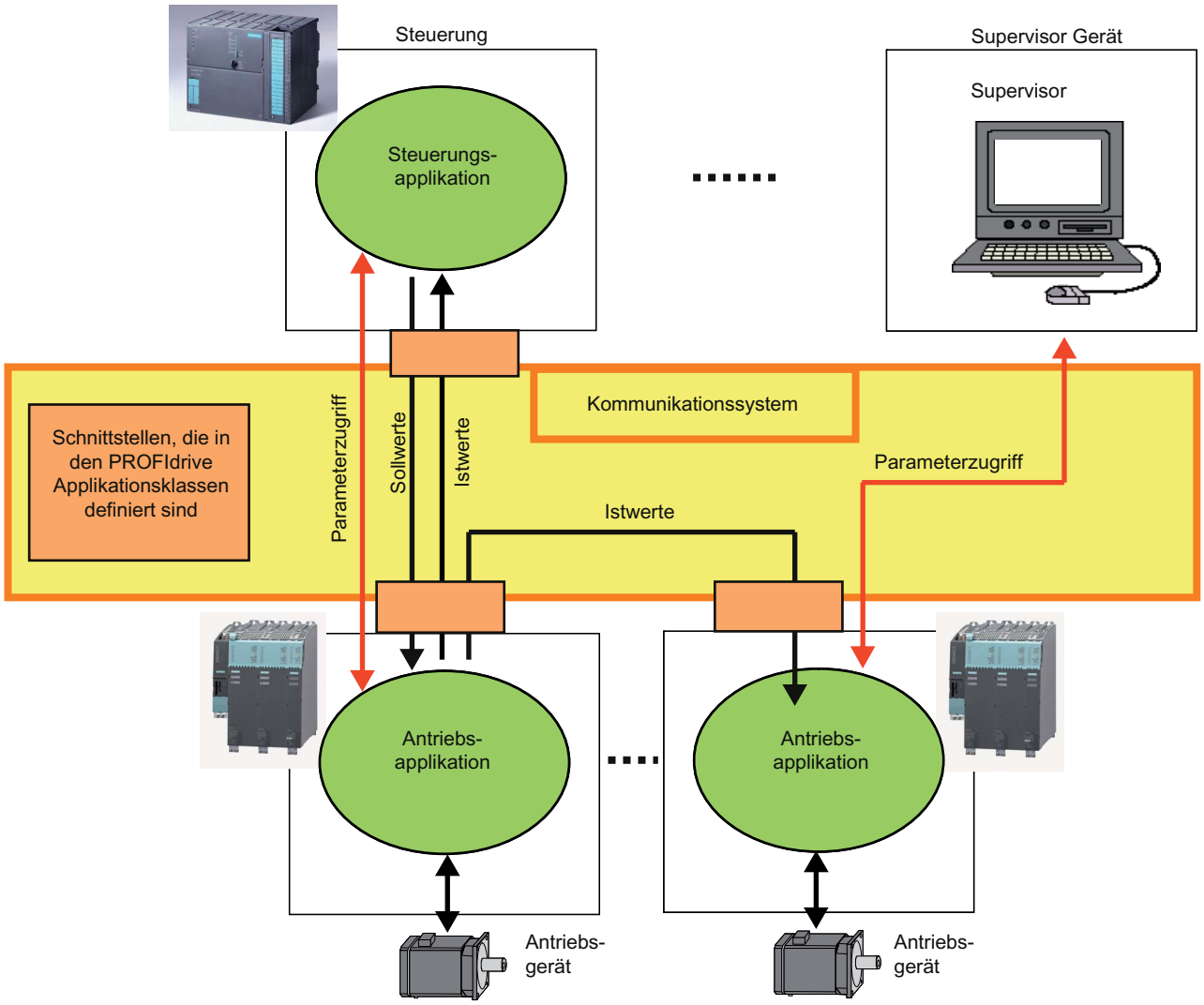


Bild 9-1 Automatisierungskonzept

Kommunikationsdienste

Im PROFIdrive Profil sind zwei Kommunikationsdienste definiert, der zyklische Datenaustausch und der azyklische Datenaustausch.

- Zyklischer Datenaustausch über zyklischen Datenkanal

  Motion Control Systeme benötigen im Betrieb zum Steuern und Regeln zyklisch aktualisierte Daten. Diese müssen über das Kommunikationssystem als Sollwerte an die Antriebsgeräte gesendet bzw. als Istwerte vom Antriebsgerät übertragen werden. Die Übertragung dieser Daten ist normalerweise zeitkritisch.

- Azyklischer Datenaustausch über azyklischen Datenkanal

  Neben dem zyklischen Datenaustausch steht außerdem ein azyklischer Parameterkanal zum Austausch von Parametern zwischen Steuerung / Supervisor und Antriebsgeräten zur Verfügung. Der Zugriff auf diese Daten ist nicht zeitkritisch.

- Alarmkanal

  Die Alarme werden ereignisgesteuert ausgegeben und zeigen das Kommen und Gehen von Fehlerzuständen an.

Die nachfolgende Grafik zeigt das Datenmodell und den Datenfluss im P-Device.

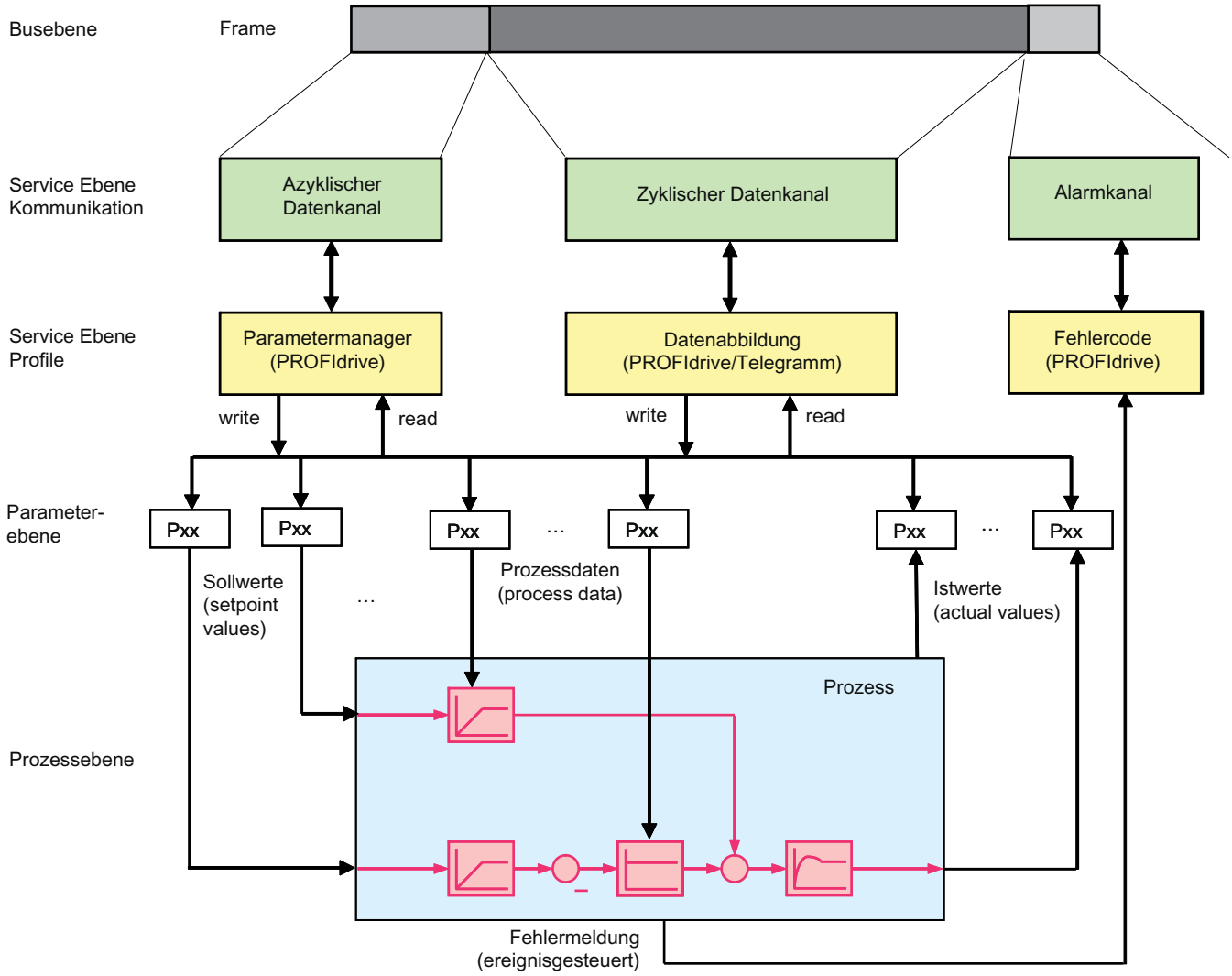


Bild 9-2 Datenmodell und Datenfluss im P-Device

Alarme und Fehlermeldungen

Die Alarme werden ereignisgesteuert ausgegeben und zeigen das Kommen und Gehen von Fehlerzuständen an.

Parameter Modell

Das Parameter Modell im PROFIdrive Profil unterscheidet zwischen den Profilparametern und den herstellerspezifischen Parametern:

- Die Profilparameter sind für Objekte definiert, die sich aus dem Gerätemodell des PROFIdrive Profils ableiten. Das sind z. B. allgemeine Funktionen wie Antriebsidentifikation, Störpuffer oder Antriebssteuerung. Diese Parameter sind für alle Antriebe gleich.
- Alle anderen Parameter sind herstellerspezifisch. Die Parameter werden nicht durch das Profil festgelegt, jedoch die Schnittstelle zum Applikationsprozess.

Der Zugriff auf die Elemente (Werte, Parameterbeschreibungen, Textelemente) eines Parameters erfolgt grundsätzlich azyklisch (mit Ausnahme von G120 Antrieben, die mit PKW zyklisch Daten austauschen können). Dazu ist eine unabhängige Request-/Response-Datenstruktur definiert.

9.4 Segmentierung in Applikationsklassen

Einbindung von Antrieben in Automatisierungslösungen

Die Einbindung von Antrieben in Automatisierungslösungen ist stark von der Antriebsaufgabe abhängig. Um die ganze, riesige Bandbreite an Antriebsanwendungen vom einfachsten Frequenzumrichter bis zu hochdynamischen, synchronisierten Mehrachssystemen in einem Profil abdecken zu können, definiert PROFIdrive sechs Anwendungsklassen, denen sich die meisten Antriebsanwendungen zuordnen lassen.

Tabelle 9- 1 Tabelle 3-1 Applikations-/Anwendungsklassen

Klasse	Antrieb
Klasse 1	Standardantriebe (wie z. B. Pumpen, Lüfter, Rührwerke etc.); realisiert in SIMOTION und SINAMICS
Klasse 2	Standardantriebe mit Technologiefunktionen
Klasse 3	Positionierantriebe; realisiert in SIMOTION und SINAMICS
Klasse 4	Motion Control Antriebe mit zentraler, übergeordneter Motion Control Intelligenz und Lageregelkonzept "Dynamic Servo Control"; realisiert in SIMOTION und SINAMICS
Klasse 5	Motion Control Antriebe mit zentraler, übergeordneter Motion Control Intelligenz und Lagersollwertschnittstelle
Klasse 6	Motion Control Antriebe mit dezentraler, in den Antrieben selber integrierter Motion Control Intelligenz

PROFIdrive definiert ein Gerätemodell aus Funktionsmodulen, die geräteintern zusammenarbeiten und die Intelligenz des Antriebssystems widerspiegeln.

Diesen Modulen sind Objekte zugeordnet, die im Profil beschrieben und hinsichtlich ihrer Funktionen definiert werden. Die gesamte Funktionalität eines Antriebs ist somit durch die Summe seiner Parameter beschrieben.

Im Gegensatz zu anderen Antriebsprofilen definiert PROFIdrive nur die Zugriffsmechanismen auf die Parameter sowie einen Subset von ca. 70 Profilparametern, wozu unter anderen z. B. Störpuffer, Antriebssteuerung und Geräteidentifikation gehören.

Alle anderen Parameter sind herstellerspezifisch, was den Antriebsherstellern große Flexibilität bei der Realisierung der Regelungsfunktionen gibt. Der Zugriff auf die Elemente eines Parameters erfolgt azyklisch über den so genannten Base Mode Parameter Access.

Hinweis

Die Aufträge mit Base Mode Parameter Access sind so kodiert, wie Sie es vielleicht von Datensatz 47 (DPV1-Kommunikation von PROFIBUS) her kennen. Für eventuelle Unterschiede, siehe Spezifika bei PROFIBUS und PROFINET IO (Seite 288).

PROFIdrive für PROFIBUS nutzt als Kommunikationsprotokoll DP-V0, DP-V1 und die DP-V2-Erweiterungen für PROFIBUS mit den darin enthaltenen Funktionen Slave-Querverkehr und Taktsynchronisation.

PROFIdrive für PROFINET enthält die Funktionen IO-Controller - IO-Device Kommunikation und Taktsynchronisation.

Anwendungsklassen

Für hochdynamische und hochkomplexe Motion Control Aufgaben ist die Anwendungsklasse 4 die wichtigste. Diese Anwendungsklasse beschreibt im Detail die Master-Slave-Beziehung zwischen dem Controller und den Antrieben, die über PROFIBUS oder PROFINET miteinander verbunden sind.

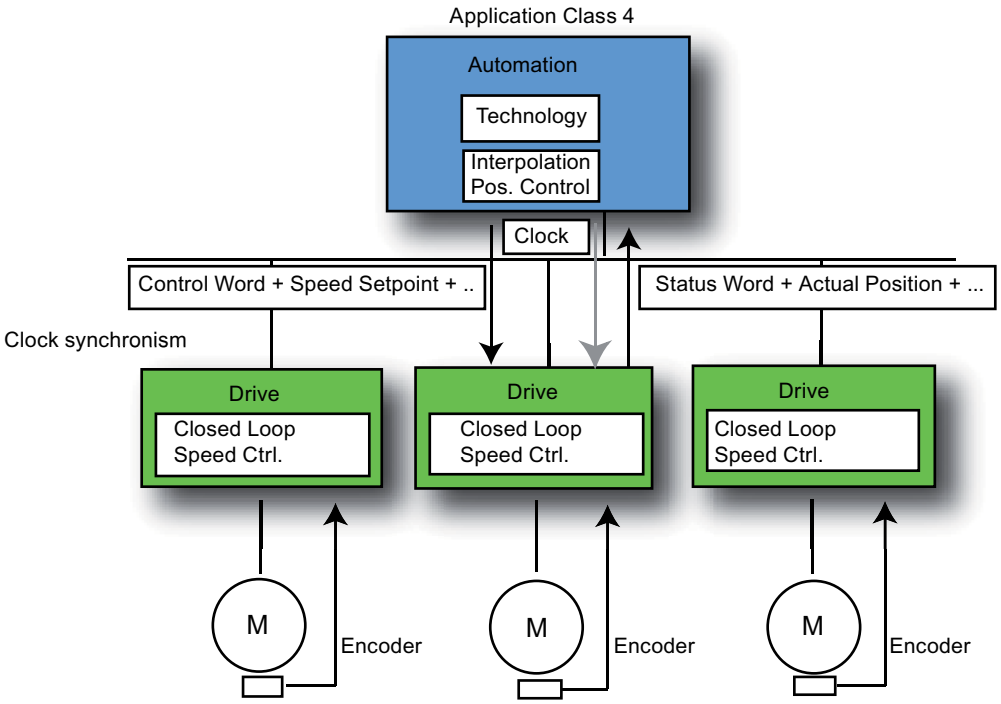


Bild 9-3 Anwendungsklassen

Durch die Funktion DSC (Dynamic Servo Control) wird die Dynamik und die Steifigkeit des Lageregelkreises deutlich verbessert. Bei SIMOTION normalerweise um die bei Drehzahl-Sollwert-Schnittstellen auftretenden Totzeiten (Sendezeit, Rechenzeit für Controller und Device), die durch ein zusätzliches, relativ einfaches Rückkoppelnetzwerk im Antrieb minimiert werden. Der Lageregler wird dabei im Antrieb vom SIMOTION-Controller über eine Vorsteuerung und einer Positionsabweichung vorgesteuert, was sehr schnelle Lagereglertakte (z. B. bei Servo 125 µs im SINAMICS S) ermöglicht, und Totzeiten ausschließlich auf das Führungsverhalten reduziert.

9.5 PROFIdrive-spezifische Datentypen

Beschreibung

Zur Verwendung der PROFIdrive-konformen Kommunikation ist eine Reihe von Datentypen definiert. Detaillierte Informationen dazu finden Sie in folgenden Normen:

- IEC 61800-7-203
- IEC 61800-7-303
- IEC 61158-5

In diesen Normen finden Sie die Datentypen ausführlich definiert. Nachfolgend werden die wichtigsten Datentypen aufgelistet. Die Datentypen werden z. B. von der Funktion _readDriveParameterDescription geliefert.

Hinweis

Für die S7 Kommunikation bzw. die Kommunikation mit SINAMICS müssen Sie mit der Systemfunktion *AnyType_to_BigByteArray* bzw. *BigByteArray_to_AnyType* eine Typkonvertierung bei verschiedenen Datentypen (Normalisierter Wert N2, N4; Normalisierter Wert X2, X4; Festkommawert E2 und Zeitkonstanten T2 und T4) durchführen.

PROFIdrive-Profil-spezifische Datentypen

Im PROFIdrive Profil verwendete Datentypen	Definition	Codierung (dez)
Boolean	Boolean (IEC 61158-5)	1
Integer8	Integer8 (IEC 61158-5)	2
Integer16	Integer16 (IEC 61158-5)	3
Integer32	Integer32 (IEC 61158-5)	4
Unsigned8	Unsigned8 (IEC 61158-5)	5
Unsigned16	Unsigned16 (IEC 61158-5)	6
Unsigned32	Unsigned32 (IEC 61158-5)	7
FloatingPoint32	Float32 (IEC 61158-5)	8
FloatingPoint64	Float64 (IEC 61158-5)	15

Im PROFIdrive Profil verwendete Datentypen	Definition	Codierung (dez)
VisibleString	VisibleString (IEC 61158-5)	9
OctetString	OctetString (IEC 61158-5)	10
TimeOfDay (mit Datumsanzeige)	TimeOfDay (IEC 61158-5)	11
TimeDifference	TimeDifference (IEC 61158-5)	12
Date	Date (IEC 61158-5)	13
TimeOfDay (without data indication)	TimeOfDay (IEC 61158-5)	52
TimeDifference (with data indication)	TimeDifference (IEC 61158-5)	53
TimeDifference (without data indication)	TimeDifference (IEC 61158-5)	54
Spezifische Datentypen	**Beschreibung siehe unten**	
N2 (Normalisierter Wert (16 Bit)		113
N4 (Normalisierter Wert (32 Bit)		114
V2 Bit Sequenz		115
L2 Nibble		116
R2 Reziproke Zeitkonstante		117
T2 Zeitkonstante (16 Bit)		118
T4 Zeitkonstante (32 Bit)		119
D2 Zeitkonstante		120
E2 Festkommawert (16 Bit)		121
C4 Festkommawert (32 Bit)		122
X2 Normalisierter Wert, variabel (16 Bit)		123
X4 Normalisierter Wert, variabel (32 Bit)		124

Normalisierter Wert N2, N4

Linear normalisierter Wert, 0% korrespondiert mit 0 (0x0), 100% korrespondiert mit 2^{12} (0x4000) für N2, oder 2^{28} (0x40000000) für N4. Die Länge beträgt 2 bzw. 4 Oktetts.

Codierung

Darstellung in 2er Komplementen, MSB (Most Signifikant Bit) ist das erste Bit nach dem Vorzeichen-Bit (SN) des ersten Oktetts.

- SN = 0; positive Zahlen mit 0
- SN = 1; negative Zahlen

Wertebereich N2, N4	Auflösung N2, N4	Cod. N2, N4 (Dez)	Oktett	Bit							
				8	7	6	5	4	3	2	1
-200% ≤ i ≤ (200-2^{-14})%	2^{-12} = 0,0061%	113	1	SN	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
			2	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}
-200% ≤ i ≤ (200-2^{30})%	2^{-28} = 9,3 * 10^{-8}%	114	3	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}
			4	2^{-23}	2^{-24}	2^{-25}	2^{-26}	2^{-27}	2^{-28}	2^{-29}	2^{-30}

Normalisierter Wert X2, X4 (Beispiel X=12/28)

Linear normalisierter Wert, 0% korrespondiert mit 0 (0x0), 100% korrespondiert mit 2^x. Diese Strukturen sind identisch zu N2 und N4, nur dass die Normalisierung variabel ist. Die Normalisierung kann aus den Parameterbeschreibungen ermittelt werden. Die Länge beträgt 2 bzw. 4 Oktetts.

Codierung

Darstellung in 2er Komplementen, MSB (Most Significant Bit) ist das erste Bit nach dem Vorzeichen-Bit (SN) des ersten Oktetts.

- SN = 0; positive Zahlen mit 0
- SN = 1; negative Zahlen

Wertebereich X2, X4	Auflösung X2, X4	Cod. X2, X4 (Dez)	Oktett	Bit							
				8	7	6	5	4	3	2	1
-800% ≤ i ≤ 800-2^{-12})%	2^{-12}	123	1	SN	2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}
			2	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}
-800% ≤ i ≤ 800-2^{-28})%	2^{-28}	124	3	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}
			4	2^{-21}	2^{-22}	2^{-23}	2^{-24}	2^{-25}	2^{-26}	2^{-27}	2^{-28}

Festkommawert E2

Linearer Festkommawert mit vier Stellen nach dem Dezimalpunkt. 0 korrespondiert mit 0 (0x0), 128 korrespondiert mit 2^{14} (0x4000). Die Länge beträgt 2 Oktetts.

Codierung

Darstellung in 2er Komplementen, MSB (Most Signifikant Bit) ist das erste Bit nach dem Vorzeichen-Bit (SN) des ersten Oktetts.

- SN = 0; positive Zahlen mit 0
- SN = 1; negative Zahlen

Wertebereich E2	Auflösung	Cod (dez)	Oktett	Bit							
				8	7	6	5	4	3	2	1
-256+2^{-7} ≤ i ≤ 256-2^{-7}	2^{-7} = 0,0078125	121	1	SN	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}
			2	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}

Festkommawert C4

Linearer Festkommawert mit vier Stellen nach dem Dezimalpunkt. 0 korrespondiert mit 0 (0x0), 0,0001 korrespondiert mit 2^0 (0x0000 0001).

Codierung

Wie bei Integer32, die Wichtung der Bits wurde um den Faktor 10000 reduziert.

Wertebereich	Auflösung	Codierung (dez)	Länge
-214748,3648 ≤ i ≤ 214748,3648	10^{-4} = 00001	122	4 Oktett

Bitsequenz V2

Bitsequenz zur Kontrolle und Darstellung von Applikationsfunktionen. 16 Boolesche Variablen sind zu 2 Oktetts kombiniert.

Wertebereich	Auflösung	Cod (dez)	Oktett	Bit							
				8	7	6	5	4	3	2	1
		115	1	15	14	13	12	11	10	9	8
			2	7	6	5	4	3	2	1	0

Nibble (Halbbyte) L2

Vier zusammengehörige Bits ergeben ein Nibble. Vier Nibbles sind durch zwei Oktetts repräsentiert.

Codierung

Wertebereich	Auflösung	Cod (dez)	Oktett	Bit							
				8	7	6	5	4	3	2	1
-	-	116	1	Nibble 3				Nibble 2			
			2	Nibble 1				Nibble 0			

Zeitkonstanten T2 und T4

Zeitdaten als Vielfaches der Abtastzeit T_a. Interpretierter Wert = Interner Wert * T_a

Codierung

- T2: Wie bei Unsigned16 mit eingeschränktem Wertebereich 0 ≤ x ≤ 32767
  Wenn interpretiert, werden interne Werte, die außerhalb des Wertebereichs liegen, auf 0 gesetzt.
- T4: Wie bei Unsigned32

Die Werte von Zeitparametern der Typen D2, T2, T4 und R2 beziehen sich immer auf die spezifizierte konstante Abtastzeit T_a. Die zugehörige Abtastzeit (Parameter p0962) wird für die Interpretation des internen Wertes benötigt.

Wertebereich	Auflösung	Codierung (dez)	Länge
0 ≤ i ≤ 32767 * T_a	T_a	118	2 Oktett
0 ≤ i ≤ 4294967295 * T_a	T_a	119	4 Oktett

Zeitkonstante D2

Zeitdaten als Bruchteil der konstanten Abtastzeit T_a. Interpretierter Wert = Interner Wert * T_a/16348

Codierung

- T2: Wie bei Unsigned16 mit eingeschränktem Wertebereich 0 ≤ x ≤ 32767
  Wenn interpretiert, werden interne Werte, die außerhalb des Wertebereichs liegen, auf 0 gesetzt.

Wertebereich	Auflösung	Codierung (dez)	Länge
0 ≤ i ≤ (2-2-14) * T_a	T_a	120	2 Oktett

Zeitkonstante R2

Zeitdaten als reziprokes Vielfaches der konstanten Abtastzeit T_a. Interpretierter Wert = 16348 * T_a/ Interner Wert

Codierung

- T2: Wie bei Unsigned16 mit eingeschränktem Wertebereich 0 ≤ x ≤ 16384
  Wenn interpretiert, werden interne Werte, die außerhalb des Wertebereichs liegen, auf 16384 gesetzt.

Wertebereich	Auflösung	Codierung (dez)	Länge
1 * Ta ≤ i ≤ 16384 * T_a	T_a	117	2 Oktett

Hinweis

Weitere Datentypen:

Standard Datentypen PROFIBUS/PROFINET (nur in englisch verfügbar) (Seite 321)

Profilspezifische Datentypen PROFIBUS/PROFINET (nur in englisch verfügbar) (Seite 332)

Siehe auch

Parameter-Request-/Response Datensatz (Seite 284)

9.6 Azyklische Kommunikation (Base Mode Parameter Access)

9.6.1 Azyklische Kommunikation

Beschreibung

PROFIdrive-Antriebsgeräte werden mit Steuer-Signalen und Soll-Werten von der Steuerung versorgt und liefern Status-Signale und Ist-Werte.

Diese Signale werden normalerweise zyklisch (d. h. ständig) zwischen Steuerung und Antrieb übertragen.

Daneben kennen PROFIdrive-Antriebsgeräte Parameter, in denen weitere benötigte Daten gehalten werden, etwa Fehler-Codes, Warnungen, Reglerparameter, Motordaten, ... Diese Daten werden normalerweise nicht zyklisch (d. h. ständig) übertragen, sondern nur bei Bedarf "azyklisch". Auch Befehle an den Antrieb können durch Parameterzugriffe übermittelt werden.

Grundsätzlich erfolgt das Schreiben/Lesen von Parametern aus PROFIdrive-Antrieben azyklisch über den so genannten "Base Mode Parameter Access". Der "Base Mode Parameter Access" kann sowohl mit PROFIBUS als auch mit PROFINET benutzt werden. Für die Unterschiede zwischen PROFIBUS und PROFINET, siehe Spezifika bei PROFIBUS und PROFINET IO (Seite 288).

Dieser Dienst wird von PROFIdrive definiert und bereitgestellt und kann parallel zu der zyklischen Kommunikation auf dem jeweiligen Bus zum Einsatz kommen. Das PROFIdrive Profil legt fest, wie genau dieser grundlegenden Mechanismus genutzt wird für den Schreib-Lesezugriff auf Parameter eines PROFIdrive-konformen Antriebs.

9.6.2 Parameter lesen und schreiben mit Base Mode Parameter Access

Beschreibung

Die Kommunikation zum Schreiben/Lesen von Parametern erfolgt bei PROFIdrive-Antrieben, wie z. B. SINAMICS S120, demnach immer über den Base Mode Parameter Access, dessen Aufbau im PROFIdrive Profil definiert ist. Der Aufbau findet sich z. B. auch im SINAMICS S120 Funktionshandbuch unter **Azyklische Kommunikation**.

Ein Parameterzugriff besteht dabei immer aus einem Pärchen:

- Write Request ("Datensatz Schreiben")
- Read Request ("Datensatz Lesen")

Die Reihenfolge muss eingehalten werden, egal ob lesender oder schreibender Zugriff.

Mit einem "Datensatz Schreiben" wird der Parameterauftrag übermittelt (z. B. einen Parameter x lesen). Mit einem "Datensatz Lesen", wird die Antwort auf diesen Parameterauftrag abgeholt (Wert des Parameters x).

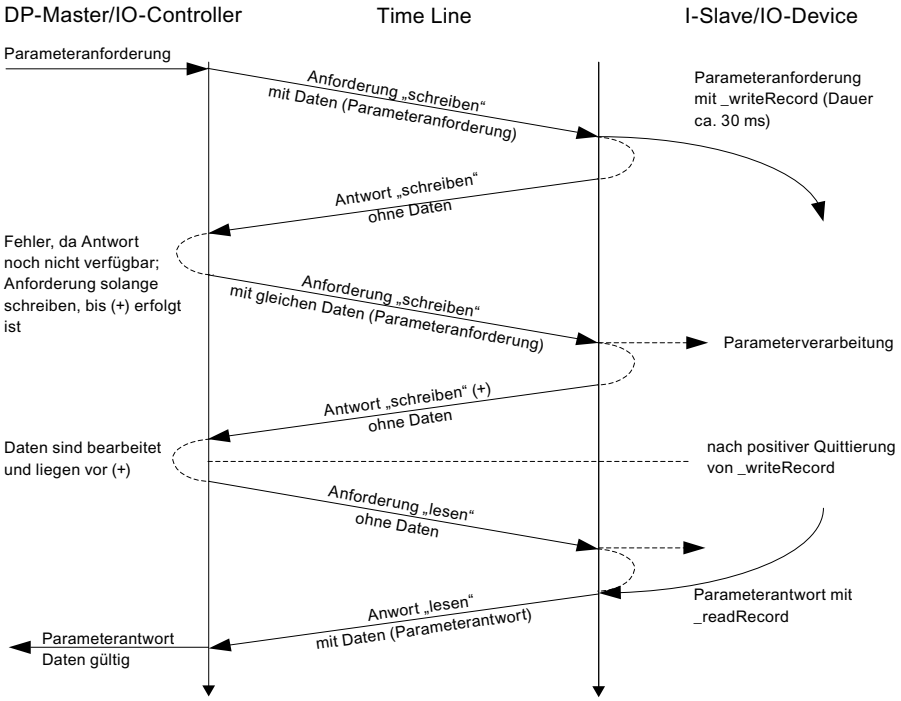


Bild 9-4 Azyklisch lesen und schreiben

Aus der Abbildung **Azyklisch lesen und schreiben** geht hervor, dass sowohl "Datensatz Schreiben" als auch "Datensatz Lesen" jeweils aus folgenden Teilen besteht:

- Request ("Anforderung")
- Response ("Antwort")

Die Steuerung wertet diese "Request Reference" nicht aus. Das Anwenderprogramm kann bzw. sollte diese Referenz jedoch auswerten.

Schreiben von Parametersätzen

Zum Schreiben von Parameterwerten (einer oder mehrere) werden zunächst der Auftragsstruktur Daten (P-Request/Response Datensatz) übergeben, die dann per "Datensatz Schreiben" mit _writeRecord übertragen werden. Durch wiederholtes "Datensatz lesen" (_writeRecord ohne Daten) kann der Status überprüft werden, bis eine positive Quittierung kommt. Danach wird _readRecord ("Datensatz lesen") solange gesendet, bis der Slave die Daten liefert.

Hinweis

Ein "Datensatz Schreiben" ohne Daten dient dazu den Status von "Datensatz schreiben" mit Daten zu ermitteln, bis die positive Quittierung erfolgt.

Eine erfolgreiche Beendigung von "Datensatz Schreiben" signalisiert nur die fehlerfreie Übertragung des Datensatzes über den Kommunikationsweg, aber nicht die fehlerfreie Ausführung des Vorgangs im Zielgerät.

Lesen von Parametersätzen

Zum Lesen von Parameterwerten wird zunächst der Datenblock zusammengestellt, welche(r) Parameter gelesen werden soll(en). Dieser Datensatz wird über das Paar "Datensatz Schreiben"-"Datensatz Lesen" (zuerst _writeRecord und dann _readRecord) an den Antrieb übertragen. Ein nachfolgendes "Datensatz Lesen" liefert dann **einmalig** die angeforderten Werte (die gleiche Auftragsreferenz wird in der Antwort auch mitgeliefert).

Die Abläufe finden Sie oben auch bildlich dargestellt.

Im PROFIdrive Profil wird festgelegt, wie Daten übertragen werden, die größer als 1 Byte sind. Es gilt das sog. "Big Endian"-Format, das die höchstwertigen Teile zuerst überträgt:

WORD		High Byte (Byte 1)	Low Byte (Byte 2)
DOUBLE WORD	High Word	High Byte (Byte 1)	Low Byte (Byte 2)
	Low Word	High Byte (Byte 3)	Low Byte (Byte 4)

Darstellung WORD und DWORD im Big Endian Format

Da die Steuerung ggf. andere interne Datenrepräsentationen hat, ist es dann erforderlich, dass bei der Zusammenstellung und Auswertung der Daten im P-Request-/Response Datenblock (Datensatz 47) eine Konvertierung explizit vorgenommen wird.

Bei SIMOTION ist eventuell eine Konvertierung erforderlich, siehe Programmbeispiel.

Siehe auch

Regel 5 - Maximal acht Aufrufe gleichzeitig in SIMOTION möglich (Seite 302)

Parameter-Request-/Response Datensatz (Seite 284)

9.6.3 Parameter-Request-/Response Datensatz

Aufbau des P-Request -/Response Datensatzes

Der besteht grundsätzlich aus:

- Einem Header (Auftragskennung, Ziel-Achse/Drive-Objekt, Anzahl Parameter im Auftrag)
- Einer Request Reference; Auftragsreferenz, um einen Auftrag zu identifizieren
- Auftragsdaten (Attribut, Anzahl Elemente/Indices, Parameternummer und -subindex), bei Schreibaufträgen auch die Werte.
- Werten, die der Funktion übergeben werden

Für einen WRITE- bzw. READ-Request haben die übertragenen Daten folgenden Aufbau.

	Parameter Auftrag	Byte n+1	Byte	Offset
Header	Request-Header	Request Reference	RequestID	0
		Achse	Anzahl Parameter	2

	Parameter Auftrag	Byte n+1	Byte	Offset
Details	1. Parameter	Attribut	Anzahl Elemente	4
		Parameternummer		6
		Subindex		8
	...			
	n. Parameter	Attribut	Anzahl Elemente	
		Parameternummer		
		Subindex		
Werte nur beim Schreiben	1. Parameterwert(e)	Format	Anzahl Werte	
		Werte		
	...			
	n. Parameterwert(e)	Format	Anzahl Werte	
		Werte		
		...		

Aufbau nach Base Mode Parameter Access - Parameter Request

Für den anschließenden Parameter Response ist folgender Aufbau festgelegt. Dieser muss mit _readRecord abgeholt werden.

	Parameter Antwort	Byte n+1	Byte	Offset
	Request-Header	Request Reference gespiegelt	RequestID gespiegelt oder Fehler	0
		Achs-Nr / DO-ID gespiegelt	Anzahl Parameter	2
Werte nur beim Lesen	1. Parameterwert(e)	Format	Anzahl Werte	4
		Werte oder Fehlercodes		6
		...		
	...			
Fehlerwerte nur bei negativer Antwort	n. Parameterwert(e)	Format	Anzahl Werte	
		Werte oder Fehlercodes		
		...		

Aufbau nach Base Mode Parameter Access - Parameter Response

Die genaue Codierung der einzelnen Bestandteile der Datenstruktur kann dem PROFIdrive Profil oder dem Funktionshandbuch SINAMICS S120 entnommen werden. Wichtig ist die Zuordnung von "Request" und "Response" sowie "Datensatz Schreiben" und "Datensatz Lesen" über die Auftragsreferenz "Request Reference" in obigerTabelle.

Request Reference

Die "Request Reference" wird für die Zuordnung von Write Request zu nachfolgendem Read Request, da prinzipiell die Steuerung mehrere Vorgänge parallel (bis zu 8) zu unterschiedlichen Zielgeräten über denselben Feldbus abwickeln kann.

Beschreibung der Felder Parameterauftrag und -antwort

Feld	Datentyp	Werte	Bemerkung
Auftragsreferenz	Unsigned8	0x01 ... 0xFF	
	Eindeutige Identifizierung des Auftrag-/Antwortpaares für den Master. Der Master ändert die Auftragsreferenz mit jedem neuen Auftrag. Der Slave spiegelt die Auftragsreferenz in seiner Antwort.		
Auftragskennung	Unsigned8	0x01 0x02	Leseauftrag Schreibauftrag
	Gibt an, um welchen Auftrag es sich handelt. Beim Schreibauftrag werden die Änderungen im flüchtigen Speicher (RAM) ausgeführt. Zur Übernahme der geänderten Daten in den nichtflüchtigen Speicher muss ein Speichervorgang ausgeführt werden (p0971, p0977).		
Antwortkennung	Unsigned8	0x01 0x02 0x81 0x82	Leseauftrag (+) Schreibauftrag (+) Leseauftrag (-) Schreibauftrag (-)
	Spiegelung der Auftragskennung mit der Zusatzinformation, ob der Auftrag positiv oder negativ ausgeführt wurde. Negativ bedeutet: Der Auftrag konnte ganz oder teilweise nicht ausgeführt werden. Es werden pro Teilantwort statt der Werte die Fehlerwerte übertragen.		
Antriebsobjekt–Nummer	Unsigned8	0x01 ... 0xFE	Nummer
	Vorgabe der Antriebsobjekt–Nummer bei einem Antriebsgerät mit mehreren Antriebsobjekten. Es kann über die gleiche DPV1-Verbindung auf verschiedene Antriebsobjekte mit je einem eigenen Parameternummernbereich zugegriffen werden.		
Anzahl Parameter	Unsigned8	0x01 ... 0x27	Anzahl 1 ... 39 Begrenzt durch DPV1-Telegrammlänge
	Definiert bei Multiparameterauftrag die Anzahl der folgenden Bereiche Parameteradresse und/oder Parameterwert. Für einfache Aufträge ist Anzahl Parameter = 1.		
Attribut	Unsigned8	0x10 0x20 0x30	Wert Beschreibung Text (Nicht implementiert bei SINAMICS)
	Art des Parameterelements, auf das zugegriffen wird.		
Anzahl Elemente	Unsigned8	0x00 0x01 ... 0x75	Sonderfunktion Anzahl 1 ... 117 Begrenzt durch DPV1-Telegrammlänge
	Anzahl der Arrayelemente, auf die zugegriffen wird.		
Parameternummer	Unsigned16	0x0001 ... 0xFFFF	Nummer 1 ... 65535

Feld	Datentyp	Werte	Bemerkung
	Adressiert den Parameter, auf den zugegriffen wird.		
Subindex	Unsigned16	0x0000 ... 0xFFFE	Nummer 0 ... 65534
	Adressiert das erste Arrayelement des Parameters, auf das zugegriffen wird.		
Format	Unsigned8	0x02 0x03 0x04 0x05 0x06 0x07 0x08 Andere Werte 0x40 0x41 0x42 0x43 0x44	Datentyp Integer8 Datentyp Integer16 Datentyp Integer32 Datentyp Unsigned8 Datentyp Unsigned16 Datentyp Unsigned32 Datentyp FloatingPoint Siehe PROFIdrive Profile V3.1 Zero (ohne Werte als positive Teilantwort eines Schreibauftrags) Byte Word Double word Error
	Format und Anzahl spezifizieren den nachfolgend durch Werte belegten Platz im Telegramm. Beim Schreibvorgang sind bevorzugt Datentypen nach PROFIdrive Profile anzugeben. Ersatzweise sind auch Byte, Wort und Doppelwort möglich.		
Anzahl Werte	Unsigned8	0x00 ... 0xEA	Anzahl 0 ... 234 Begrenzt durch DPV1-Telegrammlänge
	Gibt die Anzahl der folgenden Werte an.		
Fehlerwerte	Unsigned16	0x0000 ... 0x00FF	Bedeutung der Fehlerwerte --> siehe Tabelle 4-29
	Die Fehlerwerte bei negativer Antwort. Wenn die Werte aus einer ungeraden Anzahl von Bytes bestehen, wird ein Nullbyte angehängt. Damit wird die Wortstruktur des Telegramms sichergestellt.		
Werte	Unsigned16	0x0000 ... 0x00FF	
	Die Werte des Parameters beim Lesen oder Schreiben. Wenn die Werte aus einer ungeraden Anzahl von Bytes bestehen, wird ein Nullbyte angehängt. Damit wird die Wortstruktur des Telegramms sichergestellt.		

Für weitere Informationen zur Codierung von PROFIdrive Datentypen, siehe PROFIdrive-spezifische Datentypen (Seite 277)

Siehe auch

Programmbeispiel (Seite 316)

Parameter lesen und schreiben mit Base Mode Parameter Access (Seite 282)

9.6.4 Spezifika bei PROFIBUS und PROFINET IO

Globale und lokale Parameter

Nach PROFIdrive werden zwei Parameterbereiche unterschieden:

- Globale Parameter; diese sind dem ganzen Antriebsgerät zugeordnet. Wenn Sie verschiedene DOs eines Antriebsgerätes adressieren möchten, zeigt ein globaler Parameter immer denselben Wert.
- Lokale Parameter; diese Parameter sind achs- bzw. DO-spezifisch. Die achs- und DO-spezifischen Parameter können für jedes Achs-DO verschiedene Werte haben.

Entsprechend leiten sich die beiden Zugriffsarten für den Base Mode Parameter Access ab:

- Base Mode Parameter Access - Local (BMPL)
- Base Mode Parameter Access - Global (BMPG)

Spezifische Eigenschaften einer azyklischen Kommunikation bei PROFIBUS

Für Kommunikation über PROFIBUS wird der Datensatz 47 (0x002F) zum Zugriff auf Parameter in PROFIdrive Antrieben verwendet.

- Base Mode Parameter Access – Global; die Parameter (aller DOs, globale und lokale Parameter) des Antriebsgerätes sind über die PAPs des Antriebsgerätes ansprechbar.

Hinweis

Die PROFIdrive Norm legt fest, dass in PROFIdrive Antrieben kein Pipelining von Aufträgen unterstützt wird, dass also daher zu einem Antriebsgerät immer nur ein "Datensatz Schreiben/Lesen" gleichzeitig möglich ist. Sind aber mehrere PROFIdrive-Antriebsgeräte an einer Steuerung über PROFIBUS angeschlossen, so kann zu jedem dieser Antriebsgeräte je ein Auftrag zeitlich parallel abgewickelt werden. Die Höchstanzahl ist dann steuerungsabhängig. Daten für SIMOTION sind in Regel 5 - Maximal acht Aufrufe gleichzeitig in SIMOTION möglich (Seite 302) angegeben.

Spezifische Eigenschaften einer azyklischen Kommunikation bei PROFINET IO

Bei Verwendung von PROFINET ändert sich an den grundlegenden Abläufen nichts, jedoch lautet die Datensatz-Nummer dann 0xB02E ("Base Mode Parameter Access - Local") und 0XB02F("Base Mode Parameter Access - Global").

Auf globale Parameter eines Antriebes kann grundsätzlich sowohl bei BMPL als auch bei BMPG über jeden Parameterzugangspunkt (PAP/MAP) des Antriebes zugegriffen werden.

Beim **BMPL** erfolgt beim Zugriff auf Lokale Parameter (DO) automatisch ein Addressierung auf die lokalen Parameter des DOs dem der PAP/MAP zugeordnet ist. Eine Angabe der DO-ID in der Parameter request Struktur ist deshalb nicht notwendig bzw. wird von Antrieb ignoriert.

Use Case:
Jede Achse (DO) hat ihren eigenen Parameterzugangspunkt (zwangläufig so bei Einachsern). Die Auswahl der Achse erfolgt über die logische Adresse, der Anwender muss sich nicht um die Verwaltung der DO-IDs kümmern.

Beim **BMPG** kann über den PAP/MAP des DOs neben den DO-eigenen lokalen Parametern auch auf lokale Parameter anderer DOs zugegriffen werden. Die Adressierung des gewünschten DOs erfolgt über die DO-ID in der Parameter request Struktur. D.h. beim BMPG muss immer eine gültige DO-ID mit übergeben werden.

Use Case:
Wenn der Zugriff auf die Parameter eines Mehrachser Drive nur über einen PAP/MAP (logische Adresse erfolgen soll) bzw. wenn Parameter auf DOs gelesen werden sollen, die keinen eigenen MAP/PAP haben (insbesondere PROFIBUS Geräte). Dafür muss sich der Anwender beim BMPG um die Verwaltung der DO-IDs kümmern.

9.6.5 Fehlerauswertung

Beschreibung

Es können zwei unterschiedliche Arten von Fehlern im Zusammenhang mit den Base Mode Parameter Access Diensten auftreten:

- Fehler in der Kommunikation (Übermittlung der Daten)

  Beispielsweise könnte das adressierte Gerät nicht vorhanden oder eingeschaltet sein. Diese Art von Fehlern wird über Rückgabewerte der Systemfunktionen übermittelt und kann der Beschreibung der Systemfunktionen in den SIMOTION-Referenzlisten entnommen werden.

- Fehler bei der Bearbeitung der Aufträge selber

  Beispielsweise könnte versucht werden, auf einen Read-Only-Parameter schreibend zuzugreifen.

  Fehlercodes zu dieser zweiten Art von Fehlern sind für PROFIdrive-konforme Antriebe in der PROFIdrive-Norm festgelegt und sind unten aufgelistet.

  Durch die Response ID 0x81 (hex) bzw. 0x82 (hex) wird ein Fehler beim Parameterzugriff gekennzeichnet.

  Die Fehlercodes werden in der Antwort des Antriebsgerätes im P-Response-/Request Datenblock zurückgeliefert, siehe Tabelle unten. Die Unterscheidung, ob es sich um einen Fehler-Code oder um einen "echten" Wert eines angefragten Parameters handelt, kann man in der Parameter Response dem Feld "Format" entnehmen, siehe Tabelle Aufbau nach Base Mode Parameter Access - Parameter Response (Seite 284), Offset 4, "Format".

  Die Codierung für das Feld "Format" können Sie ebenfalls dort entnehmen. Code 0x44 (hex) deutet auf einen Fehler-Code im Feld "Werte" hin. Andere Werte von "Format" legen fest, in welchem Zahlenformat (z. B. Bool, Byte, Integer8, ...) der Wert im Feld "Werte" zurückgeliefert wurde.

Hinweis

Die Fehlercodes bis 0x19 entsprechen dem PROFIdrive Profil. Die Fehlercodes ab 0x65 sind herstellerspezifisch und können deshalb von Antrieb zu Antrieb unterschiedlich sein.

Fehlercodes in Base Modes Parameter Access Antworten

Fehler-code	Bedeutung	Bemerkung	Zusatz-Info
0x00	Unzulässige Parameternummer.	Zugriff auf nicht vorhandenen Parameter.	–
0x01	Parameterwert nicht änderbar.	Änderungszugriff auf einen nicht änderbaren Parameterwert.	Subindex
0x02	Untere oder obere Wertgrenze überschritten.	Änderungszugriff mit Wert außerhalb der Wertgrenzen.	Subindex
0x03	Fehlerhafter Subindex.	Zugriff auf nicht vorhandenen Subindex.	Subindex
0x04	Kein Array.	Zugriff mit Subindex auf nicht indizierten Parameter.	–

Fehler-code	Bedeutung	Bemerkung	Zusatz-Info
0x05	Falscher Datentyp.	Änderungszugriff mit Wert, der nicht zum Datentyp des Parameters passt.	–
0x06	Kein Setzen erlaubt (nur zurücksetzen).	Änderungszugriff mit Wert ungleich 0, wo dies nicht erlaubt ist.	Subindex
0x07	Beschreibungselement nicht änderbar.	Änderungszugriff auf nicht änderbares Beschreibungselement.	Subindex
0x09	Beschreibungsdaten nicht vorhanden.	Zugriff auf nicht vorhandene Beschreibung (Parameterwert ist vorhanden).	–
0x0B	Keine Bedienhoheit.	Änderungszugriff bei fehlender Bedienhoheit.	–
0x0F	Kein Textarray vorhanden	Zugriff auf nicht vorhandenes Textarray (Parameterwert ist vorhanden).	–
0x11	Auftrag wegen Betriebszustand nicht ausführbar.	Zugriff ist aus nicht näher spezifizierten temporären Gründen nicht möglich.	–
0x14	Wert unzulässig.	Änderungszugriff mit Wert, der zwar innerhalb der Grenzen liegt, aber aus anderen dauerhaften Gründen unzulässig ist (Parameter mit definierten Einzelwerten).	Subindex
0x15	Antwort zu lang.	Die Länge der aktuellen Antwort überschreitet die maximal übertragbare Länge.	–
0x16	Parameteradresse unzulässig.	Unzulässiger oder nicht unterstützter Wert für Attribut, Anzahl Elemente, Parameternummer oder Subindex oder eine Kombination.	–
0x17	Format unzulässig.	Schreibauftrag: Unzulässiges oder nicht unterstütztes Format der Parameterdaten.	–
0x18	Anzahl Werte nicht konsistent.	Schreibauftrag: Anzahl Werte der Parameterdaten passen nicht mit Anzahl Elemente in der Parameteradresse zusammen.	–
0x19	Antriebsobjekt existiert nicht.	Zugriff auf ein Antriebsobjekt, das nicht existiert.	–
0x65	Parameter momentan deaktiviert.	Zugriff auf einen Parameter, der zwar vorhanden ist, aber zum Zeitpunkt des Zugriffes keine Funktion erfüllt (z. B. n-Regelung eingestellt und Zugriff auf Parameter von U/f–Steuerung).	–
0x6B	Parameter %s [%s]: Kein Schreibzugriff bei freigegebenem Regler.	–	–
0x6C	Parameter %s [%s]: Unbekannte Einheit.	–	–
0x6D	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Geber (p0010 = 4).	–	–
0x6E	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Motor (p0010 = 3).	–	–
0x6F	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Leistungsteil (p0010 = 2).	–	–
0x70	Parameter %s [%s]: Schreibzugriff nur in Schnellinbetriebnahme (p0010 = 1).	–	–

Fehler-code	Bedeutung	Bemerkung	Zusatz-Info
0x71	Parameter %s [%s]: Schreibzugriff nur in Bereit (p0010 = 0).	–	–
0x72	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Parameter-Reset (p0010 = 30).	–	–
0x73	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Safety (p0010 = 95).	–	–
0x74	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Tech. Applikation/Einheiten (p0010 = 5).	–	–
0x75	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand (p0010 ungleich 0).	–	–
0x76	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Download (p0010 = 29).	–	–
0x77	Parameter %s [%s] darf im Download nicht geschrieben werden.	–	–
0x78	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Antriebskonfiguration (Gerät: p0009 = 3).	–	–
0x79	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Festlegung Antriebstyp (Gerät: p0009 = 2).	–	–
0x7A	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Datensatzbasis–Konfiguration (Gerät: p0009 = 4).	–	–
0x7B	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Geräte-Konfiguration (Gerät: p0009 = 1).	–	–
0x7C	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Geräte-Download (Gerät: p0009 = 29).	–	–
0x7D	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Geräte-Parameter-Reset (Gerät: p0009 = 30).	–	–
0x7E	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Gerät bereit (Gerät: p0009 = 0).	–	–
0x7F	Parameter %s [%s]: Schreibzugriff nur in Inbetriebnahmezustand Gerät (Gerät: p0009 ungleich 0).	–	–

Fehler-code	Bedeutung	Bemerkung	Zusatz-Info
0x81	Parameter %s [%s] darf im Download nicht geschrieben werden.	–	–
0x82	Übernahme der Steuerungshoheit ist über BI: p0806 gesperrt.	–	–
0x83	Parameter %s [%s]: Gewünschte BICO-Verschaltung unmöglich.	BICO-Ausgang liefert nicht Float-Wert, der BICO–Eingang benötigt aber Float.	–
0x84	Parameter %s [%s]: Parameteränderung gesperrt (siehe p0300, p0400, p0922)	–	–
0x85	Parameter %s [%s]: Keine Zugriffsmethode definiert.	–	–
0xC8	Unterhalb aktuell gültiger Grenze.	Änderungsauftrag auf einen Wert, der zwar innerhalb der "absoluten" Grenzen liegt, der aber unterhalb der aktuell gültigen unteren Grenze liegt.	–
0xC9	Oberhalb aktuell gültiger Grenze.	Änderungsauftrag auf einen Wert, der zwar innerhalb der "absoluten" Grenzen liegt, der aber oberhalb der aktuell gültigen oberen Grenze liegt (z. B. vorgegeben durch die vorliegende Umrichterleistung).	–
0xCC	Schreibzugriff nicht erlaubt.	Schreibzugriff nicht erlaubt, da Zugriffsschlüssel nicht vorhanden.	–

9.6.6 Zusatzinformationen zu den Parametern eines PROFIdrive-Antriebs

Beschreibung

Aus einem PROFIdrive Antriebsgerät kann man nicht nur die Werte von Parametern lesen, sondern auch Beschreibungen zu den Parametern.

Die Unterscheidung wird bei der Übermittlung des "Parameter Request" im P-Response-/Request Datenblock im Feld "Attribute" vorgenommen:

Attribute = 0x10 (hex)	Wert
Attribute = 0x20 (hex)	Parameter-Beschreibung "Parameter Description"
Attribute 0 0x30 (hex)	Parameter-Text

Wird statt des Wertes zu einem Parameter dessen "Parameter Description" angefordert, enthält das Feld "Value" in der "Parameter Response" die Beschreibung (Datentyp, ggf. Anzahl der Indices des Parameters, ...).

Hinweis

In der Regel sind Parameter-Beschreibungen nur lesbar.

9.6.7 Systembefehle in SIMOTION

9.6.7.1 SIMOTION Systembefehle _writeRecord/_readRecord

Beschreibung

Ein "Datensatz Schreiben" kann in SIMOTION mit dem Systembefehl _writeRecord() erfolgen. Ein "Datensatz Lesen" kann mit dem Systembefehl _readRecord() erfolgen. Damit ist es also möglich, Parameter in einem PROFIdrive-Antrieb zu lesen, schreiben oder dessen Beschreibung auszulesen.

Die Beschreibung der Systemfunktionen und deren Eingangsparameter und Rückgabewerte können in der SIMOTION-Systemdokumentation gefunden werden:

- C2xx Referenzliste
- D4XX Referenzliste
- P3xx Referenzliste

Diese Systembefehle _write/_readRecord sind universell verwendbar, nicht nur für PROFIdrive-Antriebe, sondern auch für z. B. intelligente Sensoren am PROFIBUS oder andere Peripherie-Baugruppen, die die sog. DP-V1-Dienste von PROFIBUS unterstützen.

Hinweis

Bei SIMATIC lauten die entsprechenden Systemfunktionen:

SFB52 WR_REC Datensatz Schreiben

SFB53 RD_REC Datensatz Lesen

Um die SIMOTION-Systembefehle _write/_readRecord nutzen zu können, ist Folgendes notwendig:

- PROFIBUS DP: Ein Zugriff über logische I/O-Adresse wie auch Diagnose-Adresse ist möglich.
- PROFINET IO: Ein Zugriff ist nur über die Diagnose-Adresse eines Parameter Access Points (PAP) möglich.

Weiterhin ist die DO-ID nur für Datensatz 47 (0x002f) und Global Access (PROFINET 0xb02f) relevant. Bei Local Access (PROFINET IO 0xb02e) ist die Diagnoseadresse des zugehörigen PAP relevant, die DO-ID wird nicht ausgewertet.

So ist z. B. im Zusammenhang mit PROFIdrive-Antrieben die Telegramm-Anfangsadresse des zyklisch ausgetauschten PROFIdrive-Telegramms an das Gerät nötig.

Hat ein Antrieb auf einem Antriebsgerät mehrere Achsen (mit einer gemeinsam genutzten PROFIBUS-Anschaltung), so benötigt man zusätzlich zur Unterscheidung der Achsen im selben Gerät noch die Achsnummer "Axis-No." bzw. "DO-ID" im Datensatz 47. Beispiele für solche mehrachsigen Antriebe sind der SIMODRIVE 611universal bzw. SINAMICS S120. Zur Bestimmung der "DO-ID" bei SINAMICS S120, siehe SINAMICS S120-Inbetriebnahmehandbuch **Azyklische Kommunikation**.

Über "Axis-No." bzw. "DO-ID" = 0 kann auf die so genannten "globalen Parameter" zugegriffen werden. Beispiele für solche "globalen Parameter" sind:

- P0918: PROFIBUS Adresse
- P0964: Device Identifikation (u. a. Hersteller, Version, Anzahl der Achsen)
- P0965: Profil Nummer (welche PROFIdrive Version ist implementiert)
- P0978: Liste der DO-Ids (welche "Axis-No." bzw. "DO-ID" wurde eingestellt)

9.6.7.2 SIMOTION Systembefehle _writeDrive.../_readDrive...

Beschreibung

Während die Systemfunktionen _readRecord und _writeRecord universell einsetzbar sind für alle Geräte an PROFIBUS, die die sog. DP-V1-Dienste "Datensatz Schreiben/Lesen" unterstützen, sind die folgenden Befehle speziell abgestimmt auf PROFIdrive-Antriebe laut PROFIdrive Profil:

- _read/writeDriveParameter (liest/schreibt einen ggf. indizierten Antriebsparameter)
- _read/writeDriveMultiParameter (liest/schreibt mehrere ggf. indizierte Antriebsparameter zu einem Antrieb bzw. Antriebsobjekt)
- _readDriveFaults (liest den aktuellen Störpuffereintrag eines Antriebs bzw. Antriebsobjektes)
- _readDriveParameterDescription (liest die Beschreibungsdaten eines Parameter aus dem Antrieb bzw. Antriebsobjekt)
- _readDriveMultiParameterDescription (liest die Beschreibungsdaten von mehreren Parameter aus dem Antrieb bzw. Antriebsobjekt)

Die Befehle erstellen intern den für die für die einzelnen Funktionen benötigten Datensatz 47 laut PROFIdrive Profil anhand der vom Anwender beim Aufruf der Systemfunktionen übergebenen Parameter und wickeln die Kommunikation zum PROFIdrive-Antrieb über "Datensatz Schreiben/Lesen" selbstständig ab.

Die Beschreibung der Befehle kann in der SIMOTION-Systemdokumentation gefunden werden, siehe die Referenzlisten der jeweiligen Plattform.

Siehe auch

Gültigkeitsbereich für die Regeln (Seite 304)

9.6.7.3 Vergleich der Systembefehle

Beschreibung

In der folgenden Tabelle sind die wichtigsten Unterschiede zwischen den beiden Gruppen von Systembefehlen dargestellt:

Befehlsgruppe	Vorteil	Nachteil
_readRecord _writeRecord	• Allgemein verwendbar, nicht nur für DP-V1-Dienste zu Antrieben • Setzt nur die Kenntnis irgendeiner I/O-Adresse *auf dem Antriebsgerät* sowie der "DO-ID" bzw. "Axis-No" auf dem Antriebsgerät voraus	• Anwender muss Datensatz selber zusammenstellen • Anwender muss zwei Aufrufe programmieren für Parameterzugriffe in einem PROFIdrive-Antrieb • Anwender muss sich ggf. um die Konvertierung der Daten selber kümmern • "DO-ID" bzw. "Axis-No" muss bekannt sein
_readDrive... _writeDrive...	• Speziell angepasst an die typischen Kommunikationen mit PROFIdrive Antrieben • Anwender muss Aufbau des Datensatzes 47 nicht kennen • Geringerer Programmieraufwand für den Anwender bei Kommunikation zu Antrieben	• Setzt Vorhandensein bzw. Kenntnis einer I/O-Adresse *des jeweiligen Antriebsobjekts* voraus • Eine I/O-Adresse für ein Antriebsobjekt gibt es nur bei zyklischer Kommunikation (mit PROFIBUS) zu dem Antriebsobjekt, möglicherweise also z. B. nicht für I/O-Erweiterungsmodule TB30, TMxx für ausschließliche Verwendung im Antrieb • Anwender muss sich um die Konvertierung der Daten selber kümmern

Eigenschaften der Systembefehle

Die Verwendung der Antriebs-spezifischen Systembefehle _write/_readDrive... bietet für Sie einerseits mehr Komfort als die Verwendung der allgemeinen Befehle _write/_readRecord, da Sie sich nicht um den Aufbau des Datensatzes 47 kümmern müssen und nicht die aufeinander folgenden Aufrufe von _writeRecord und _readRecord jeweils in Schrittketten programmieren müssen. Da der Aufbau der übertragenen Datensätze dem System nicht bekannt ist wegen der allgemeinen Verwendbarkeit dieser Systemfunktionen, müssen Sie beim Senden und Empfangen ggf. die Konvertierung in die Darstellung laut PROFIdrive Profil selber vornehmen, siehe Programmbeispiel (Seite 316).

Die Verwendung der Befehle _write/_readDrive... ist bis SIMOTION V4.1 beschränkt auf Fälle, bei denen es einen zyklischen Datenverkehr zum jeweiligen Antriebsobjekt gibt, da diese als Eingabe-Parameter benötigt wird. Ab dieser Version ist es möglich die DO-ID bzw. Axis-No über den Parameter **doId** der Systemfunktionen zu übergeben. Damit ist es möglich mit jedem DO zu kommunizieren, auch wenn es sich um einen azyklischen Datenverkehr handelt.

Demgegenüber kann mit _write/_readRecord auch dann noch auf Antriebsobjekte zugegriffen werden, wenn kein zyklischer Datenverkehr existiert (oder dessen I/O-Adresse nicht bekannt ist in der Applikation). Dies gelingt mit _write/_readRecord deshalb, weil für das Zusammenstellen des Datensatzes 47 die explizite Kenntnis der "DO-ID" bzw. "Axis-No." ausreicht sowie die Kenntnis irgendeiner I/O-Adresse auf dem Gerät. Dies kann z. B. dann von Vorteil sein, wenn einzelne Antriebsobjekte nur Antriebs-intern verwendet werden (d. h. ohne zyklischen Telegrammverkehr zur Steuerung) oder diese bei "generischer Programmierung" nicht allgemein bekannt ist.

9.6.7.4 Löschen von _readDrive- und _writeDrive-Aufträgen

Beschreibung

Wenn Sie nicht korrekte Lese- bzw. Schreibaufträge, die z. B. mit _readDriveParameter aufgerufen wurden, abbrechen bzw. löschen möchten, können Sie folgende Funktionen verwenden:

- _abortReadWriteRecordJobs, für die Funktionen _readRecord bzw. _writeRecord
- _abortAllReadWriteDriveParameterJobs, für die Funktionen
  - _readDrive(Multi)ParameterDescription
  - _readDrive(Multi)Parameter
  - _writeDrive(Multi)Parameter
  - _readDriveFaults

Sie können die Funktionen aufrufen, ohne dass Sie die CommandID kennen, auslesen.

9.6.8 Regeln für die Anwendung von _readRecord und _writeRecord

9.6.8.1 Regel 1 - Auftrag hat eigene Auftragsreferenz

Jeder Auftrag bekommt eine eigene Auftragsreferenz

Dies ist erforderlich, um unterschiedliche Aufträge zuordnen zu können. Die Auftragsreferenz kann wieder verwendet werden, wenn die Zuordnung auch anderweitig, z. B. durch zeitliche Abfolge, klar ist.

9.6.8.2 Regel 2 - Systemfunktionen bei asynchroner Programmierung

Beschreibung

R2: Sie müssen bei asynchroner Programmierung die Systemfunktion wiederholt mit gleichen IDs aufrufen, bis zur Beendigung der Funktion ("Langläufer"). In der Abbildung **Fehlerfreier Ablauf mit den Systemfunktionen _readRecord und _writeRecord** wird die fehlerfreie Verwendung der Systemfunktionen _writeRecord und _readRecord am Beispiel der Kommunikation zu SINAMICS S120 dargestellt.

Die Kommunikation zum Schreiben und Lesen von Parametern erfolgt zum SINAMICS S120 immer über den Datensatz 47, dessen Aufbau in der Dokumentation zum SINAMICS S120 beschrieben ist, siehe Inbetriebnahmehandbuch SINAMICS S120, Azyklische Kommunikation.

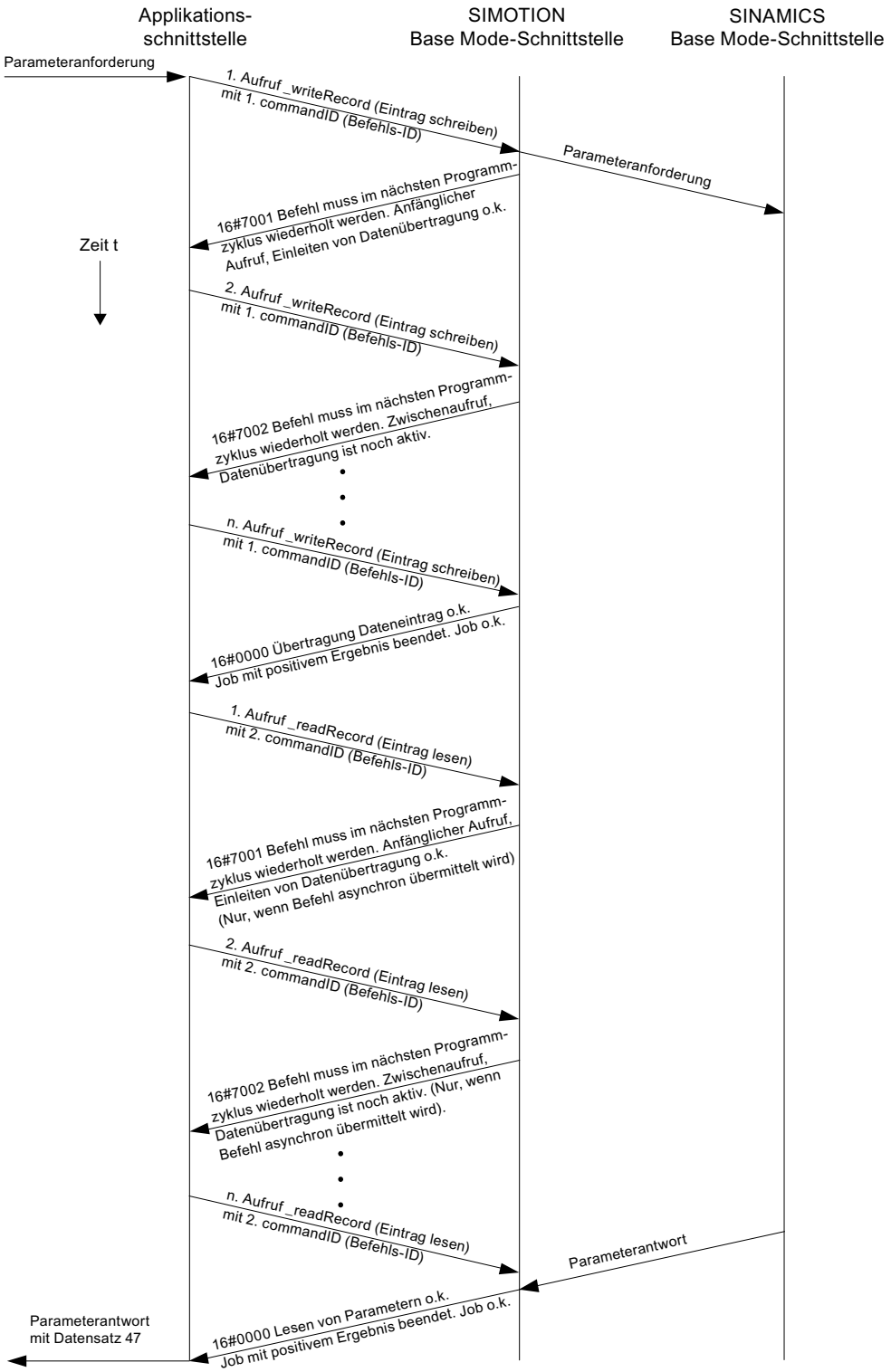


Bild 9-5 Fehlerfreier Ablauf mit den Systemfunktionen _readRecord und _writeRecord

9.6.8.3 Regel 3 - Datensatz Schreiben/Lesen pro PROFIdrive Antriebsgerät

Nur ein Datensatz Schreiben/Lesen pro PROFIdrive Antriebsgerät gleichzeitig

Im PROFIdrive Profil ist festgelegt, dass PROFIdrive-Antriebe kein Pipelining durchführen und daher nur ein Auftrag zu einer Zeit bearbeitet wird. Dies ist daher auch für SINAMICS S120 im Inbetriebnahmehandbuch so beschrieben.

Hinweis

Es ist dabei unerheblich, über welche Systemfunktionen die Übertragung in der Steuerung abgewickelt wird. Ein PROFIdrive-Antrieb kann nur einen Auftrag zu einer Zeit bearbeiten.

Hinweis

Es kann bei anderen Geräten am PROFIBUS durchaus möglich sein, dass diese mehrere "Datensatz Schreiben/Lesen" parallel unterstützen.

Hinweis

Da die Systemfunktionen _write/_readRecord universell einsetzbar sind, wird auf Steuerungsseite *nicht* verriegelt, dass nur ein "Datensatz Schreiben/Lesen" pro PROFIdrive-Antrieb zu einer Zeit angestoßen werden kann.

Folge für die Applikation auf der Steuerung:

Es muss verriegelt werden, dass die Applikation bzw. unterschiedliche Teile der Applikation gleichzeitig bzw. überlappend Aufträge an dasselbe PROFIdrive-Antriebsgerät senden, siehe auch Abschnitt unten Verriegelung von mehreren Aufrufen (Seite 305).

9.6.8.4 Regel 4 - Letzter Aufruf gewinnt bei SIMOTION

Bei SIMOTION "gewinnt" im Zweifel der letzte Aufruf

Wird die Regel 3 "Nur ein Datensatz Schreiben/Lesen pro PROFIdrive Antriebsgerät gleichzeitig" verletzt, indem zwischendurch ein zweiter _writeRecord Befehl auf denselben Antrieb abgesetzt wird, so kann die Antwort des ersten Auftrags anschließend nicht mehr gelesen werden. Der Versuch, die Antwort des Antriebs auf den ersten Auftrag zu lesen kann vom Antrieb nicht mehr bearbeitet werden und wird mit Fehler quittiert und abgebrochen. Der zeitliche Ablauf kann der Abbildung **Zweiter Aufruf von _writeRecord gewinnt im Zweifel** entnommen werden.

Um die Aufträge auf Steuerungsseite voneinander unterscheiden zu können, wurden für die Aufrufe der Systemfunktionen _writeRecord und _readRecord je getrennte commandID benutzt.

Um die Aufträge auch auf Antriebsseite voneinander unterscheiden zu können, wurden für den ersten und zweiten Auftrag eigene Auftragsreferenzen im Datensatz 47 vergeben.

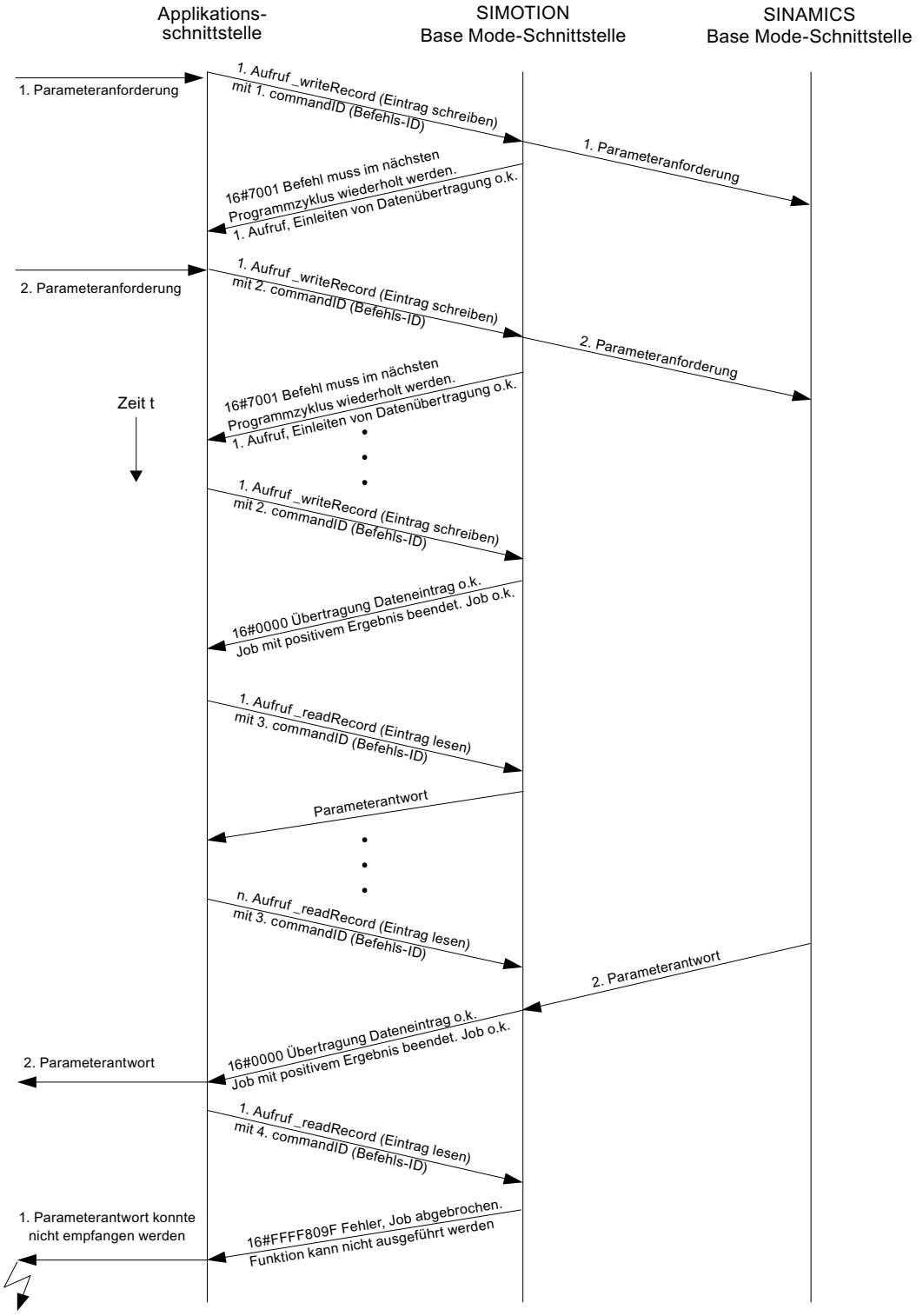


Bild 9-6 Zweiter Aufruf von _writeRecord gewinnt im Zweifel

9.6.8.5 Regel 5 - Maximal acht Aufrufe gleichzeitig in SIMOTION möglich

SIMOTION kann max. 8 Aufrufe von _write/_readRecord gleichzeitig verwalten

Obwohl laut Regel 3 (siehe Regel 3 (Seite 300)) nur ein Auftrag zu einer Zeit von *einem* PROFIdrive-Antriebsgerät gleichzeitig abgewickelt werden kann, ist es doch möglich, vom Steuerungsprogramm mehrere Aufträge parallel abzusetzen.

Dies macht bei *einem* PROFIdrive-Antrieb keinen Sinn, kann aber bei Kommunikation zu *mehreren* Antrieben parallel (oder ggf. bei anderen Geräten, die das unterstützen) sinnvoll sein.

Bei SIMOTION werden Ressourcen vorgehalten, um max. 8 Aufrufe von _write/_readRecord verwalten zu können. Die Unterscheidung erfolgt anhand der commandID von _write/_readRecord. Wird versucht, einen neunten Aufruf gleichzeitig abzusetzen, so wird dies mit Fehler von der Steuerung quittiert und unterbunden.

Der zeitliche Ablauf kann der Abbildung **8 Aufträge gleichzeitig verwalten** entnommen werden.

Es werden zunächst sieben Aufträge _writeRecord angestoßen, aber nicht beendet (keine weiteren Aufrufe von _writeRecord, um die Aufträge zum Abschluss zu bringen). Der achte Auftrag _writeRecord wird angestoßen und weiter bearbeitet bis zum Abschluss. Anschließend ist es möglich, einen neunten Aufruf abzusetzen (der hier aber wieder vom Anwenderprogramm nicht weiter bearbeitet wird). Die SIMOTION-Systemfunktion _writeRecord quittiert dann den Versuch den zehnten Auftrag abzusetzen mit Fehler 16#80C3, da dies der neunte "offene" Auftrag wäre.

Hinweis

Die Obergrenze gilt pro SIMOTION Steuerung, nicht pro Bus-Segment an der Steuerung. Es ist also unerheblich, ob die adressierten Zielgeräte an einem PROFIBUS-Segment betrieben werden, oder auf mehrere verteilt sind.

Hinweis

Da die Systemfunktionen _write/_readRecord universell einsetzbar sind, wird auf Steuerungsseite *nicht* verriegelt, dass nur ein "Datensatz Schreiben/Lesen" pro PROFIdrive-Antrieb zu einer Zeit angestoßen werden kann.

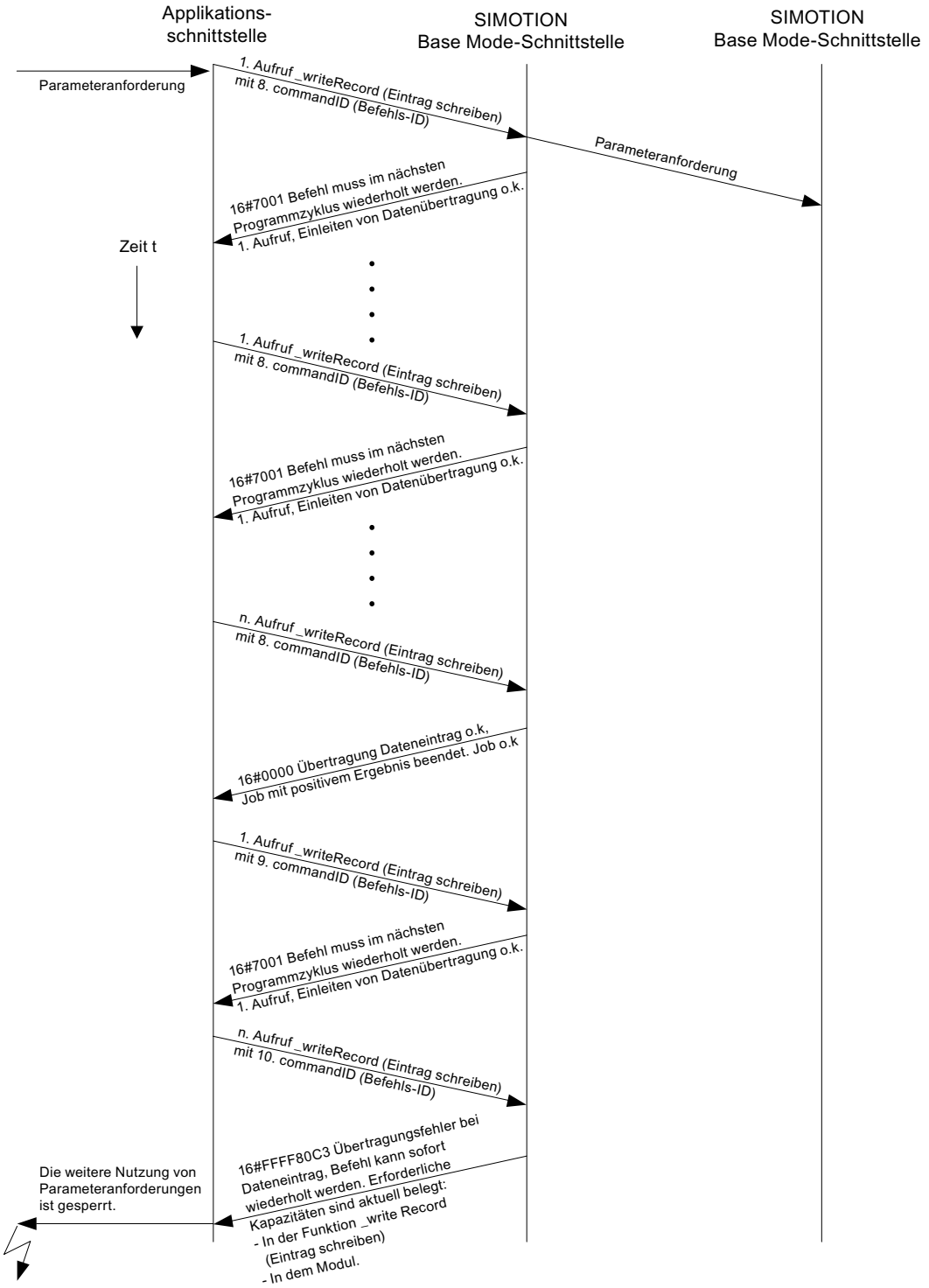


Bild 9-7 8 Aufträge gleichzeitig verwalten

Hinweis

Wenn Fehler 16#80C3 auftritt, müssen Sie die CPU auf STOP und dann wieder auf RUN setzen. Dadurch wird der Auftragspuffer gelöscht. Um den Fehler zu vermeiden, sollten Sie den Auftrag durch einen Abort-Befehl beenden, falls Sie den Auftrag nicht abschließen können.

9.6.9 Regeln für SIMOTION Befehle _writeDrive.../_readDrive...

9.6.9.1 Gültigkeitsbereich für die Regeln

Beschreibung

Die folgenden Beispiele werden am Beispiel der Systemfunktion _readDriveParameter dargestellt. Die Beschreibungen gelten aber analog auch für die anderen bereits erwähnten Systemfunktionen _writeDrive.../_readDrive... .

9.6.9.2 Regel 6 - Systemfunktion bei asynchroner Programmierung wiederholt aufrufen

Beschreibung

Der Anwender muss bei asynchroner Programmierung die Systemfunktion wiederholt aufrufen mit gleichen IDs, bis zur Beendigung der Funktion ("Langläufer").

In der folgenden Abbildung wird die fehlerfreie Verwendung der Systemfunktion _readDriveParameter dargestellt.

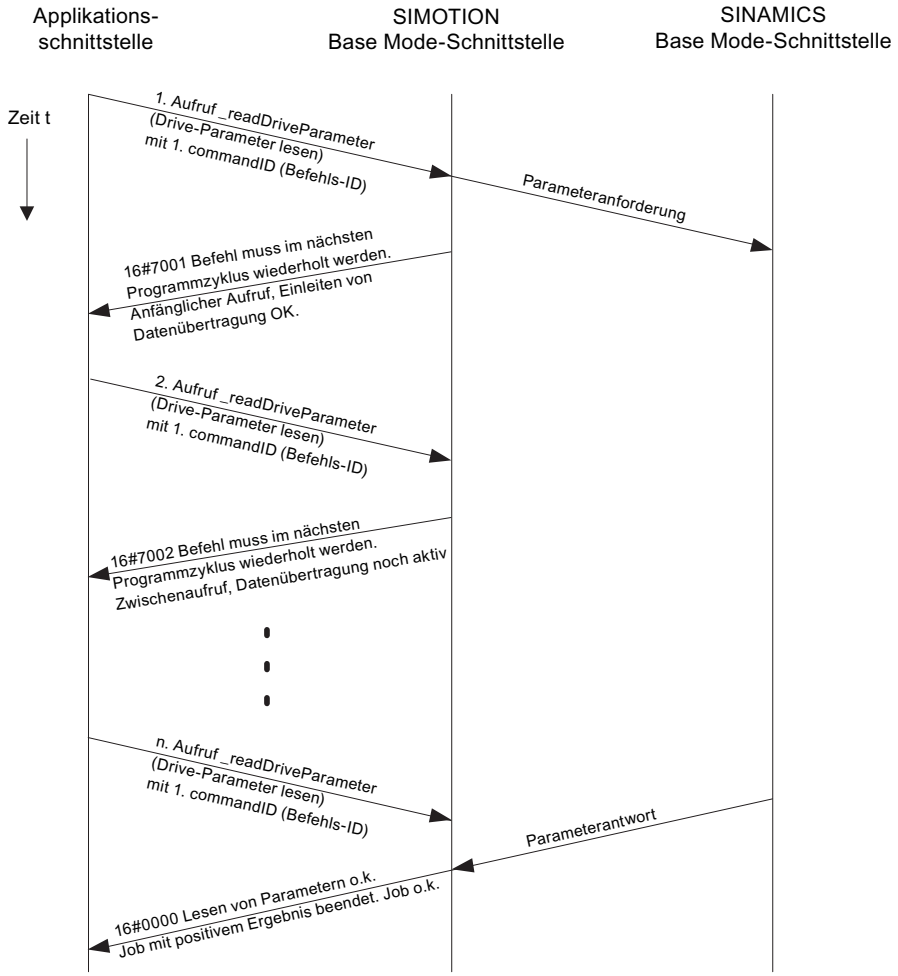


Bild 9-8 Fehlerfreier Ablauf mit den Systemfunktionen _writeDriveParameter und _readDriveParameter

9.6.9.3 Regel 7 -- Mehrere Aufrufe pro Zielgeräte gleichzeitig verriegeln

Beschreibung

In der PROFIdrive-Norm ist festgelegt, dass PROFIdrive-Antriebe kein Pipelining durchführen und daher nur ein Auftrag zu einer Zeit bearbeitet wird. Dies ist daher auch für SINAMICS S120 im Inbetriebnahmehandbuch zu SINAMICS S120 dokumentiert.

Da die SIMOTION-Systembefehle _write/_readDrive... für die häufig vorkommende Verwendung mit PROFIdrive-Antrieben erstellt wurden, wird dies bereits auf Steuerungsseite berücksichtigt.

Hinweis

Es ist dabei unerheblich, über welche Systemfunktionen die Übertragung in der Steuerung abgewickelt wird. Ein PROFIdrive-Antrieb kann nur einen Auftrag zu einer Zeit bearbeiten.

Folge für die Applikation auf der Steuerung:

Es muss verriegelt werden, dass die Applikation bzw. unterschiedliche Teile der Applikation gleichzeitig bzw. überlappend Aufträge an dasselbe PROFIdrive-Antriebsgerät senden.

Die Abbildung unten zeigt das Verhalten, wenn dies nicht berücksichtigt wird. Der Versuch, einen zweiten Auftrag (mit eigener commandID) an dasselbe Zielgerät abzusetzen, wird mit Fehler quittiert. Ein weiterer Auftrag an dasselbe Zielgerät kann erst dann wieder abgesetzt werden, wenn der erste Auftrag beendet bzw. abgebrochen wurde, siehe Abschnitt Freigabe der Verriegelung (Seite 307).

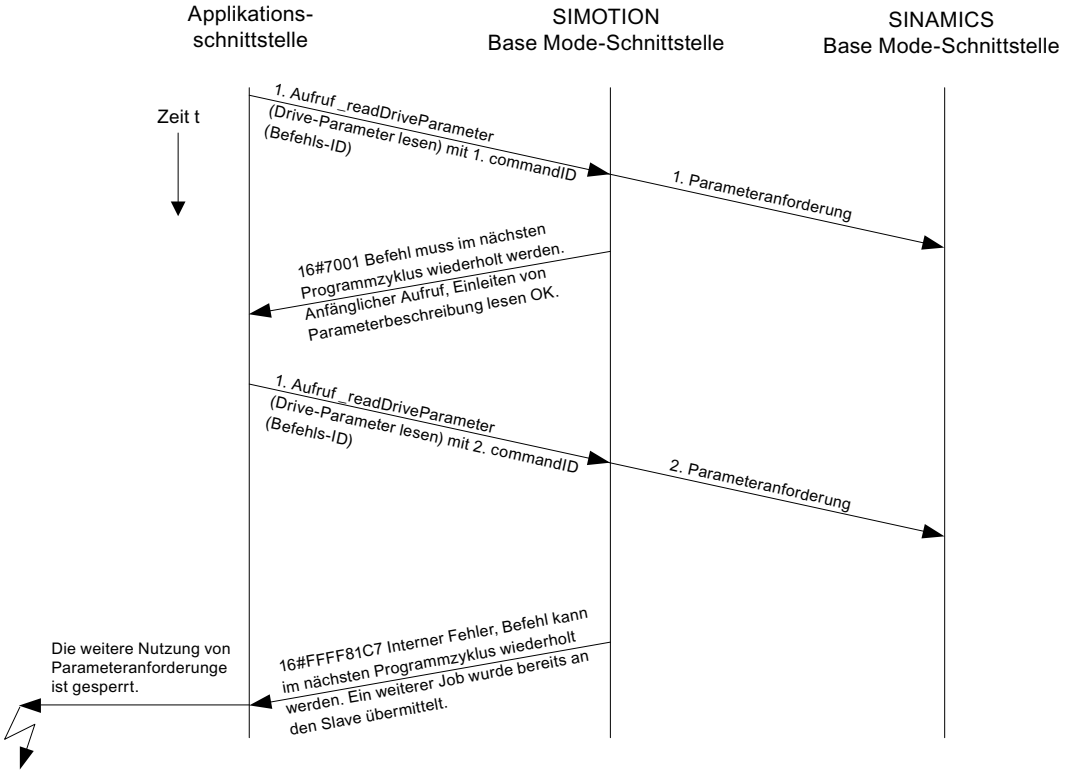


Bild 9-9 Verriegelung von mehreren Aufträgen _readDriveParameter auf ein Zielgerät

9.6.9.4 Regel 8 - Freigabe der Verriegelung nach vollständiger Abarbeitung eines Auftrags

Freigabe der Verriegelung erst bei vollständiger Abarbeitung eines Auftrags

Die folgende Abbildung zeigt, dass es nicht ausreicht, "etwas" zu warten, sondern dass man die Systemfunktionen _read/_writeDrive... wirklich so lange aufrufen muss, bis der Auftrag vollständig abgearbeitet ist. Vorher wird die Verriegelung nicht gelöst und die internen Verwaltungsressourcen frei gegeben.

Die Anzahl der Aufrufe wurde so gewählt, dass die SIMOTION DP-V1 Schnittstelle jeden Folgeaufruf für den ersten Auftrag mit 16#7002 beantwortet und somit nicht vollständig bearbeitet wird. Dies kann je nach Belastung des Busses und des Antriebs auch recht häufig erforderlich sein (>25 mal). Eine Abschätzung dafür kann nicht angegeben werden.

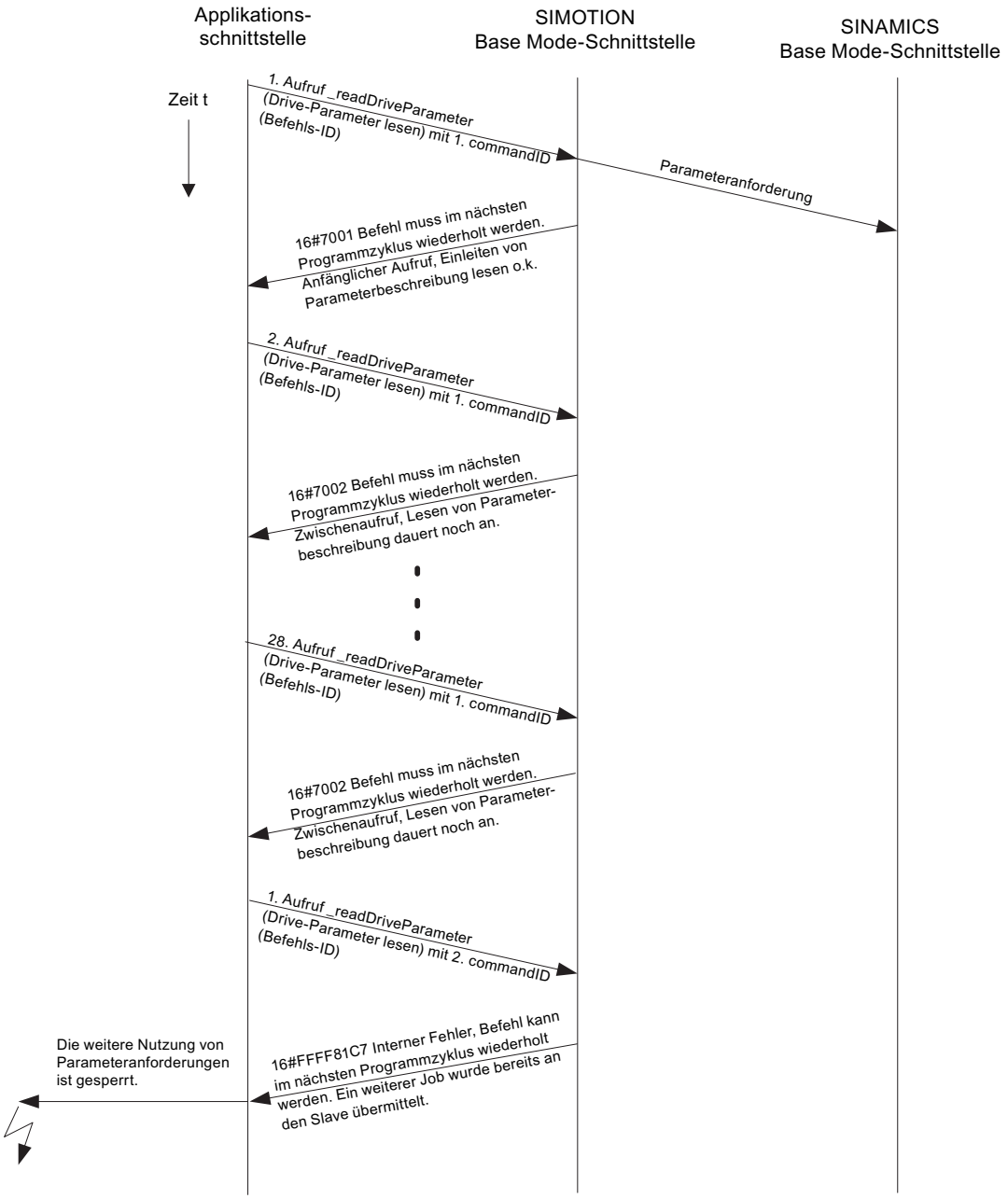


Bild 9-10 Vollständige Abarbeitung eines _readDriveParameters erforderlich zur Freigabe der Verriegelung

9.6.9.5 Regel 9 - Abbrechen von Aufträgen bei asynchronem Aufruf

CommandID wird benötigt zum Abbrechen von Aufträgen bei asynchronem Aufruf

Um den DP-V1 Dienst für das Zielgerät wieder frei zugeben, muss

- entweder der erste Auftrag beendet werden (wiederholte Aufrufe mit der commandID des ersten Auftrags)
- oder abgebrochen werden (nochmals ein Aufruf der Funktion _readDriveParameter mit der gleichen commandID wie beim ersten Anstoßen des Auftrags auch. Zusätzlich wird dazu der Eingabeparameter nextCommand mit dem Wert ABORT_CURRENT_COMMAND versehen).

Hinweis

Ab V4.1 ist ein Abbrechen ohne Kenntnis der commandID möglich, siehe Löschen von _readDrive- und _writeDrive-Aufträgen (Seite 297) .

Ein exemplarischer Aufruf der Funktion _readDriveParameter mit der ersten commandID (id1) und ABORTED_CURRENT_COMMAND sieht wie folgt aus:

```
Return_Par_read_delete :=
    readDriveParameter(
    ioId:=INPUT,
    logAddress := 256,
    parameterNumber := number,
    numberOfElements := 0,
    subIndex:= 0,
    nextCommand :=
    ABORT_CURRENT_COMMAND,
    commandId := id1);
```

Die zeitliche Abfolge kann der folgenden Abbildung entnommen werden.

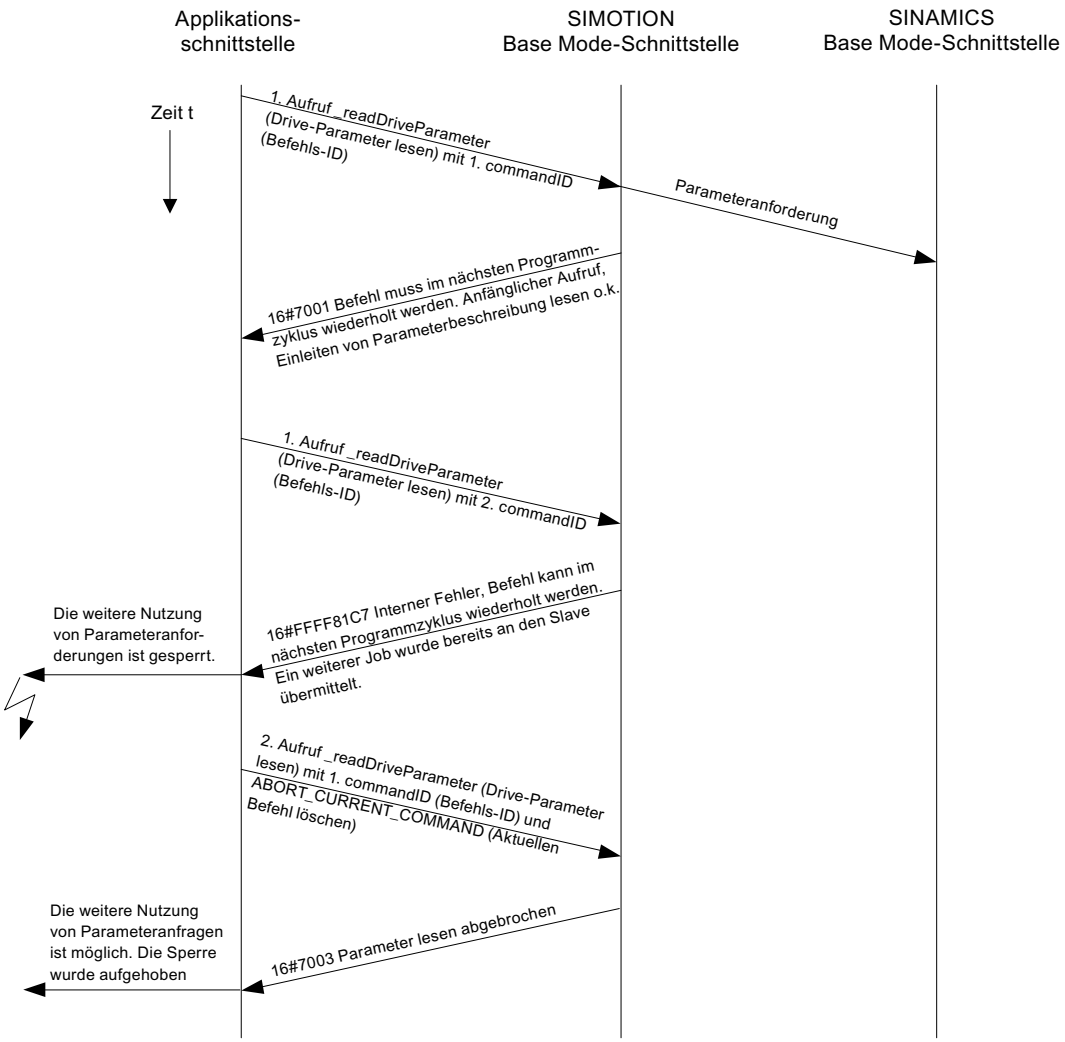


Bild 9-11 Abbrechen eines Auftrags _readDriveParameter mit bekannter commandID

Der in der folgenden Abbildung dargestellte Ablauf zeigt, dass es nicht gelingt, einen Auftrag abzubrechen ohne Kenntnis der ursprünglichen commandID. Es wird der Abbruchversuch abgebrochen, nicht der erste Auftrag. Der Grund dafür ist, dass die commandID zur Verwaltung der unterschiedlichen Aufträge im System herangezogen wird.

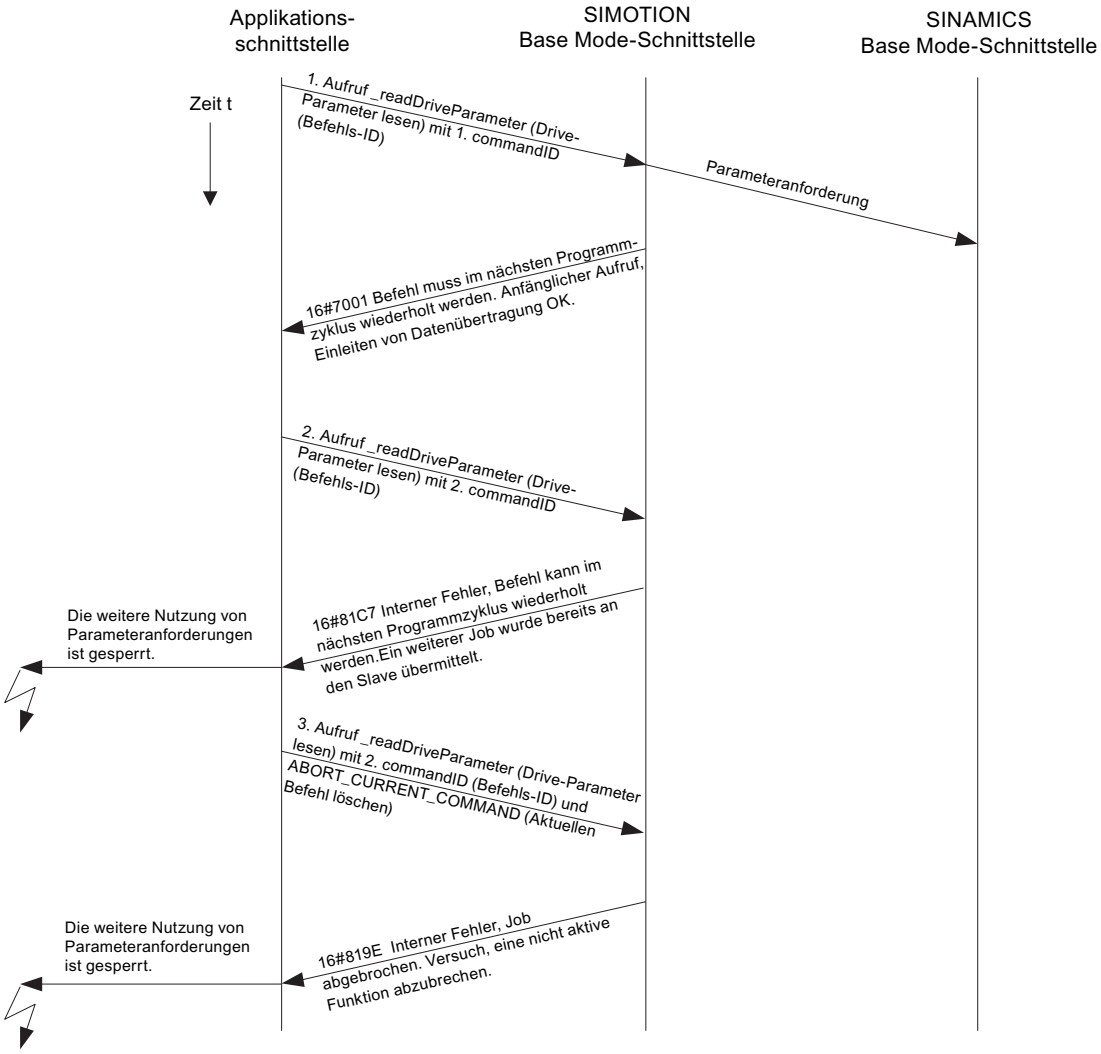


Bild 9-12 Kein Abbrechen eines Auftrags _readDriveParameter mit neuer CommandID

Hinweis

Wichtig ist also, dass das Anwenderprogramm die commandID der Aufträge so lange behält, bis der Auftrag beendet oder abgebrochen ist.

Hinweis

Besonders ist darauf zu achten, dass – z. B. durch sonstige Bedingungen gesteuert – im Anwenderprogramm nicht die Bearbeitung der _write/_readDrive... Funktionen übersprungen wird, solange diese noch nicht beendet sind.

9.6.9.6 Regel 10 - Verwaltung von 16 Aufträgen

SIMOTION verwaltet max. 16 Aufrufe für verschiedene Geräte parallel

In der Steuerung stehen begrenzte Ressourcen (Speicherplatz) zur Speicherung von Verwaltungsdaten für _write/_readDrive... Systemfunktionsaufrufe zur Verfügung. Werden zu viele Aufrufe parallel abgesetzt, so erfolgt eine Fehlermeldung, analog zur Grenze bei _read/_writeRecord im Abschnitt Maximale Anzahl von Aufrufen (Seite 302).

Bei SIMOTION werden Ressourcen vorgehalten, um max. 16 Aufrufe von _writeDrive.../_readDrive....-Systembefehlen verwalten zu können. Die Unterscheidung erfolgt anhand der commandID. Wird versucht, einen siebzehnten Aufruf gleichzeitig abzusetzen, so wird dies mit Fehler von der Steuerung quittiert und unterbunden.

9.6.9.7 Regel 11 - Parallele Aufträge unterschiedlicher Antriebsgeräte

Parallele Aufträge an unterschiedliche Antriebsgeräte sind möglich

Die **Parallele Bearbeitung von _readDriveParameter-Aufträgen** Abbildung zeigt, dass parallel Aufträge mit verschiedenen Antriebsgeräten abgewickelt werden können. Der SINAMICS Integrated einer D445 (z. B.) wird als erstes PROFIdrive-Antriebsgerät und die Erweiterungsbaugruppe CX32 als zweites PROFIdrive Antriebsgerät von einer Steuerung SIMOTION D445 genutzt.

Insgesamt werden in dem Beispiel drei Leseaufträge (zwei Aufträge auf das erste Antriebsgerät (SINAMICS Integrated) und ein Auftrag an das zweite Antriebsgerät (CX32) mittels der Systemfunktion _readDriveParameter abgesetzt.

- Der erste Leseauftrag für die SINAMICS Integrated wird absichtlich nur einmal aufgerufen, sodass die Verriegelung anspricht.
- Anschließend wird der zweite Leseauftrag auf das zweite PROFIdrive-Antriebsgerät (CX32) abgesetzt. Dieser Auftrag wird erfolgreich abgearbeitet.
- Der dritte Leseauftrag ist wieder an das erste Antriebsgerät (SINAMICS Integrated) adressiert und kann wegen des noch laufenden ersten Auftrags nicht mehr erfolgreich ausgeführt werden.

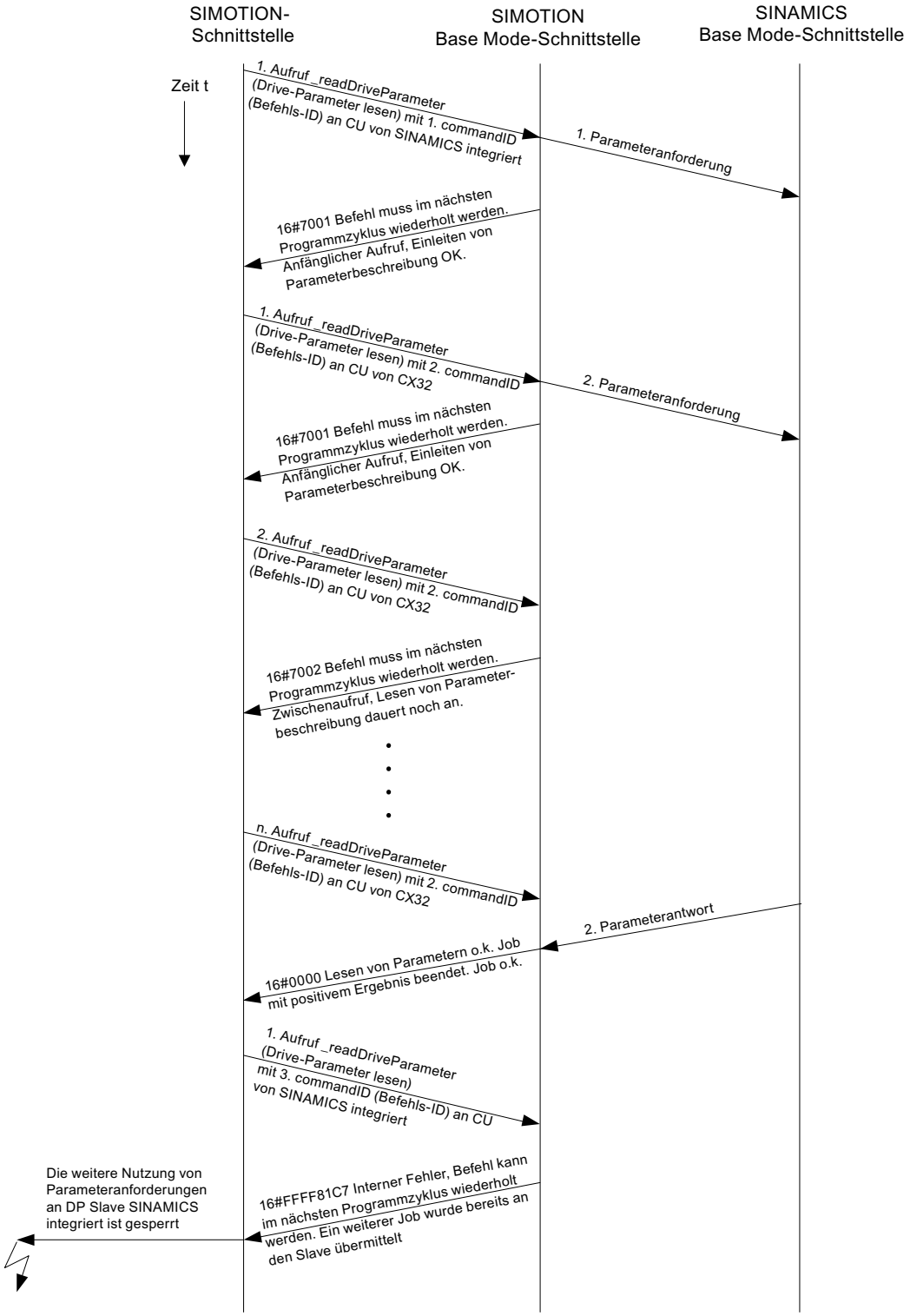


Bild 9-13 Parallele Bearbeitung von _readDriveParameter-Aufträgen an unterschiedlichen Antriebsgeräten einer Steuerung

9.6.10 Besonderheiten

9.6.10.1 Regel 12 - Datenpufferung von maximal 64 Antriebsobjekten

SIMOTION puffert die Daten von max. 64 Antriebsobjekten´

Der erste Aufruf der Funktionen _write/_readDrive... nach Systemhochlauf läuft deutlich länger als folgende Aufrufe zum gleichen Antriebsobjekt.

- Das System muss erstmalig interne Verwaltungsinformationen aufbauen, die für folgende Aufrufe zum selben Antriebsobjekt schneller zugegriffen werden können.

Es können in SIMOTION die Daten für bis zu 64 Antriebsobjekte gespeichert werden für die Verwendung mit _write/_readDrive... Dabei wird die Unterscheidung anhand der I/O-Adresse vorgenommen.

9.6.10.2 Regel 13 - Systemfunktionen können gemischt verwendet werden

Systemfunktionen _writeRecord/_readRecord und _writeDrive.../_readDrive... können gemischt verwendet werden

Eine gemischte Verwendung folgender Systembefehle ist grundsätzlich möglich:

- SIMOTION Systembefehle _writeRecord/_readRecord
- SIMOTION Systembefehle _writeDrive.../_readDrive...

Hinweis

Wichtig ist aber, dass man sich klarmacht, dass damit bei fehlender Verriegelung der Systembefehle aus beiden Befehlsgruppen mehrere Aufträge an einen PROFIdrive-Antrieb abgesetzt werden könnten (siehe folgender Abschnitt), was ein PROFIdrive-Antrieb nicht verarbeiten kann. Die systeminterne Verriegelung von _write/_readDrive... wird so umgangen.

In der Abbildung **Gemischte Verwendung von _readDrive... und _read/_writeRecord** ist dargestellt, dass insbesondere die Funktionen _write/_readRecord auch dann noch zum selben Zielgerät verwendet werden können, wenn wegen eines noch laufenden Auftrags _readDriveParameter weitere Aufträge mit demselben Befehl vom System unterbunden werden – dies ist vom Anwender zu verriegeln, da ein PROFIdrive-Antrieb dies nicht verarbeiten kann.

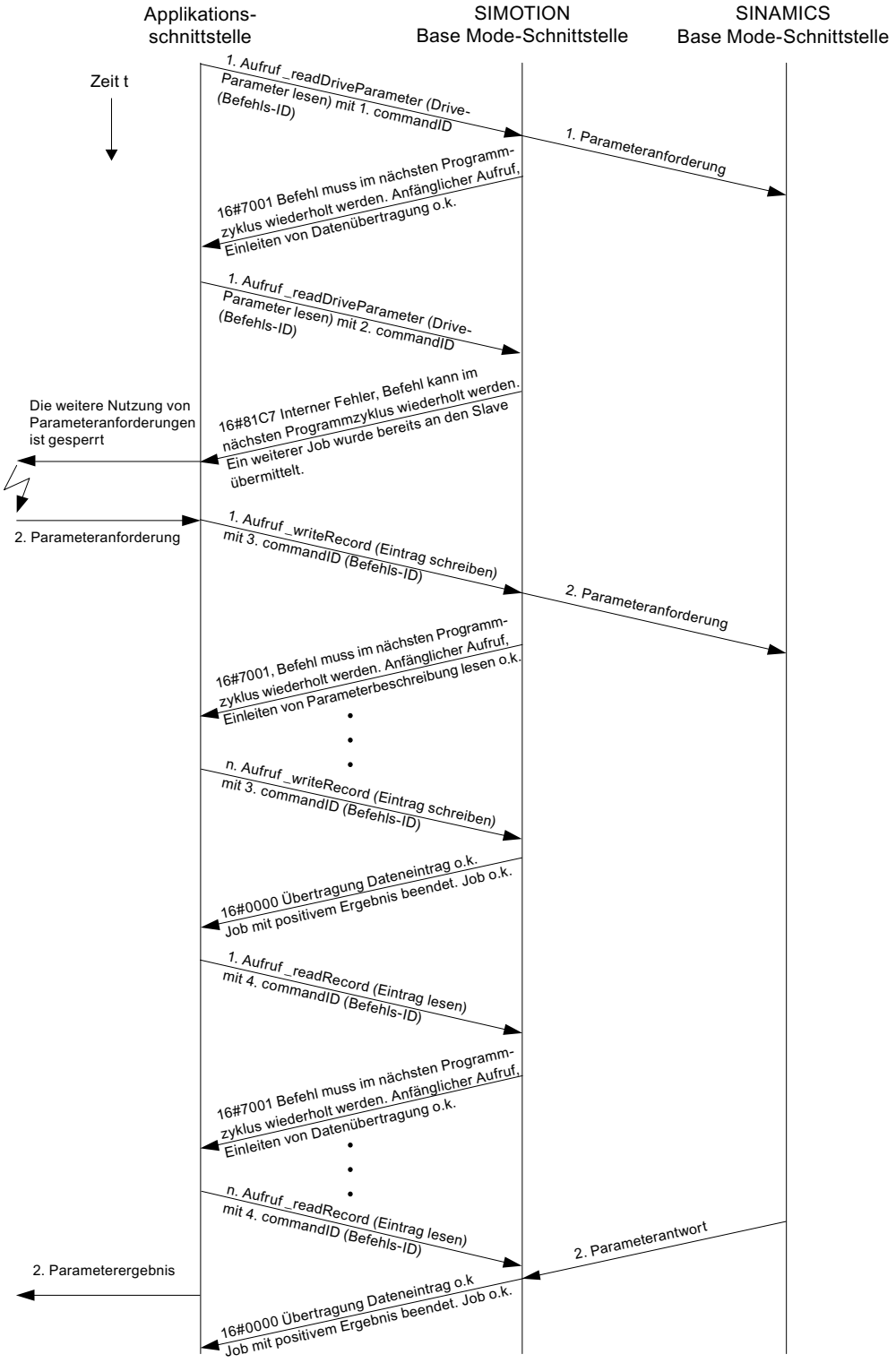


Bild 9-14 Gemischte Verwendung von _readDrive... und _read/_writeRecord

9.6.10.3 Regel 14 - Verriegelung bei gemischter Verwendung der Befehle

Anwender muss verriegeln bei gemischter Verwendung der Befehle aus den beiden Befehlsgruppen

Bei gemischter Verwendung der folgenden Systembefehle kann es dazu kommen, dass zu einem Gerät mehr als ein "Datensatz Lesen/Schreiben" gleichzeitig abgesetzt wird, da bei SIMOTION nur innerhalb der Befehlsgruppen verriegelt bzw. gepuffert wird, aber nicht untereinander.

- SIMOTION Systembefehle _writeRecord/_readRecord

und

- SIMOTION Systembefehle _writeDrive.../_readDrive...

Dies ist vom Anwender ggf. zu verriegeln, um Datenverlust/Überschneidungen zu vermeiden, da ein PROFIdrive Antrieb laut PROFIdrive Profil kein Pipelining durchführt und daher nur einen Auftrag pro Zeit bearbeiten kann.

9.6.11 Programm-Beispiele

9.6.11.1 Programmbeispiel

Beschreibung

Das nachfolgende Beispiel zeigt, wie die Systembefehle _writeRecord bzw. _readRecord verwendet werden können, um den Störcode aus Parameter p0945 eines SINAMICS-Antriebs (Antriebsobjekt DO3, I/O-Adresse 256) auszulesen.

Beispiel

Das Beispiel-Programm kann z. B. in der BackgroundTask aufgerufen werden, da die sog. "asynchrone Programmierung" verwendet wird.

```
//=============================================================================
// demonstrate reading parameter 945 (fault code) via data set 47
// using SIMOTION system functions _write/_readRecord (asynchronous call)
// INPUT address 256 is assumed to address the SINAMICS
// drive is DO3 in SINAMICS S120
//=============================================================================
INTERFACE
 PROGRAM record;
 // declare request type
 TYPE
 // declare struct of header request
 Header_Type_Request : STRUCT
    Request_Reference : USINT;
    Request_Id : USINT;
```

```
    Axis : USINT;
    Number_Of_Parameter : USINT;
 END_STRUCT;

 // declare struct of parameter address request
 Parameter_Address_Request : STRUCT
    Attribute : USINT;
    Number_Of_Elements : USINT;
    Parameter_Number : UINT;
    SubIndex : UINT;
 END_STRUCT;

 // declare struct of request
 Request : STRUCT
    Header : Header_Type_Request;
    ParameterAddress : Parameter_Address_Request;
 END_STRUCT;
// declare struct of header response
 Header_Type_Response : STRUCT
 Response_Reference : USINT;
 Response_Id : USINT;
 Axis : USINT;
 Number_Of_Parameter : USINT;
 END_STRUCT;

 // declare struct of parameter address response
 Parameter_Address_Response : STRUCT
 Format : USINT;
 Number_Of_Elements : USINT;
 Value_Or_Error_Value : DWORD; // dependent on format
 END_STRUCT

 // declare struct of response
 Response : STRUCT
 Header : Header_Type_Response;
 ParameterAdress : Parameter_Address_Response;
 END_STRUCT;
 END_TYPE
 // declare global variables
 VAR_GLOBAL
 // declare variable, that represents the dataset 47 request
 myRequest : Request;
 // declare variable, that represents the dataset 47 response
 myResponse : Response;
 // declare variable, that returns a value after calling _writeRecord
 myRetDINT : DINT;
 // declare variable, that returns a struct after calling _readRecord
 myRetstructretreadrecord : StructRetReadRecord;
 // declare array of byte,
 // which helps to create the request/response
 // with marshalling function
```

```
 bytearray : ARRAY[0..239] OF BYTE;
 // declare array of USINT,
 // because the systemfunctions _writeRecord and _readRecord
 // use this array
 usintarray : ARRAY[0..239] OF USINT;
 // declare command ids
 id_write, id_read : commandidtype;
 // declare the variable, to control step by step execution
 // start cycle with setting to 0 by user
 program_step : USINT := 3; // initially idle;
 END_VAR
END_INTERFACE
```

Implementation

```
// ===================================================================
IMPLEMENTATION
PROGRAM record
CASE program_step OF
// initialize ---------------------------------------------------
0:
 // get command ids for calling system functions
 id_write := _getcommandid();
 id_read := _getcommandid();
 // header from the request
 // here: Axis-No / DO-ID is 3
 // read Parameter 945 (drive fault code)
 myRequest.Header.Request_Reference := 16#10; // arbitrary no.
 myRequest.Header.Request_Id := 16#1; // read request
 myRequest.Header.Axis := 16#3; // axis no 3
 myRequest.Header.Number_Of_Parameter := 16#1; // one parameter

 // parameter address from the request
 myRequest.ParameterAddress.Attribute := 16#10; // read value
 myRequest.ParameterAddress.Number_Of_Elements := 16#1; // one index
 myRequest.ParameterAddress.Parameter_Number := 945; // parameter no.
 myRequest.ParameterAddress.SubIndex := 0;

 // convert myRequest to a BIBBYTEARRAY to use the marshalling functions
 // two step conversion from user defined data type
 // to usintarray type required by system functions
 bytearray := ANYTYPE_TO_BIGBYTEARRAY(myRequest,0);
 usintarray := BIGBYTEARRAY_TO_ANYTYPE(bytearray,0);

 // next step
 program_step := 1;

// execute _writeRecord -----------------------------------------
1:
```

```
 // the systemfunctions _writeRecord and _readRecord
 // have to be called in sequence.
 // the functions occur always as pair.
 // call systemfunction _writeRecord to send the request
 myRetDINT := _writerecord(
    ioid := INPUT,
    logaddress := 256, // io address
    recordnumber := 47, // data set 47 for DPV1
    offset := 0,
    datalength := 240,
    data := usintarray, //
    nextcommand := IMMEDIATELY, // use asynchronous
    commandid := id_write // use known commandID
 );
 // check the return value
 // keep calling until _writeRecord ready
 IF(myRetDINT = 0)THEN
    // next step
    program_step := 2;
 END_IF;
// wait for requested data ------------------------------------------------
// execute _readRecord
2:
 // call systemfunction _readRecord to receive the data
 myRetstructretreadrecord := _readrecord(
    ioid := INPUT,
    logaddress := 256, // io address
    recordnumber := 47, // data set 47 for DPV1
    offset := 0,
    datalength := 240,
    nextcommand := IMMEDIATELY, // use asynchronous
    commandid := id_read // use known commandID
 );
 // check the return value
 // keep calling until _readRecord ready
    IF(myRetstructretreadrecord.functionresult = 0)THEN
        // next step
        program_step := 3; // --> done
        // get data
        // two step conversion into user defined data type
        // from usintarray type given by system functions
        bytearray :=ANYTYPE_TO_BIGBYTEARRAY(
        myRetstructretreadrecord.data,0);
        myResponse := BIGBYTEARRAY_TO_ANYTYPE(bytearray,0);
        // received data can now be read from myResponse...
    END_IF;
END_CASE;
END_PROGRAM
END_IMPLEMENTATION
```

Anhang

10

10.1 Standard Datentypen PROFIBUS/PROFINET (nur in englisch verfügbar)

Subset of IEC 61158-5 standard data types for use in PROFIBUS/PROFINET profiles

Coding of most of the data types in these profile guidelines is defined in IEC 61158-6:2003, clause 6.2. However, the usage in practice is different in some cases. Thus it is highly recommended to follow the definitions and hints within this annex. The next edition of IEC 61158 is supposed to comply with the content of this annex.

Boolean

A Boolean is representing a data type that only can have two different values i.e. TRUE and FALSE. Hint: for efficiency reasons this data type is not used in application profiles.

Numeric Identifier	Data type name	Value range	Resolution	Length
1	Boolean	True/false	-	1 Octet

Bits	7	6	5	4	3	2	1	0	
True	x	x	x	x	x	x	x	x	0x01 to 0xFF
False	0	0	0	0	0	0	0	0	0x00

Integer16

An Integer16 is representing a signed number depicted by 16 bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
3	Integer16	-32768 ≤ i ≤ 32767	1	2 Octets

In two’s complement; the most significant bit (MSB) is the bit after the sign (SN) in the first Byte.

SN = 0: positive numbers and zero

SN = 1: negative numbers

Bits	7	6	5	4	3	2	1	0
Octets 1	SN	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
Octets 2	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Integer32

An Integer32 is representing a signed number depicted by 32 bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
4	Integer32	$-2^{31} \le i \le 2^{31}-1$	1	4 Octets

In two's complement; the most significant bit (MSB) is the bit after the sign (SN) in the first Byte.

SN = 0: positive numbers and zero

SN = 1: negative numbers

Bits	7	6	5	4	3	2	1	0
Octets 1	SN	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
Octets 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
Octets 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
Octets 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Integer64

An Integer64 is representing a signed number depicted by 64 bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
55	Integer64	$-2^{62} \le i \le 2^{62}-1$	1	8 Octets

In two's complement; the most significant bit (MSB) is the bit after the sign (SN) in the first Byte.

SN = 0: positive numbers and zero

SN = 1: negative numbers

Bits	7	6	5	4	3	2	1	0
Octets 1	SN	2^{62}	2^{61}	2^{60}	2^{59}	2^{58}	2^{57}	2^{56}
Octets 1	2^{55}	2^{54}	2^{53}	2^{52}	2^{51}	2^{50}	2^{49}	2^{48}
Octets 1	2^{47}	2^{46}	2^{45}	2^{44}	2^{43}	2^{52}	2^{41}	2^{40}
Octets 1	2^{39}	2^{38}	2^{37}	2^{36}	2^{35}	2^{34}	2^{33}	2^{32}
Octets 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
Octets 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
Octets 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
Octets 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Unsigned8

An Unsigned8 is representing an unsigned number depicted by 8 bits ("enumerated"). Some application profiles (e.g. PROFIsafe) are using this data type for the coding of individual single bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
5	Unsigned8	0 ≤ i ≤ 255	1	1 Octet

Tabelle 10- 1 Enumerated:

Bits	7	6	5	4	3	2	1	0
Octets 1	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Tabelle 10- 2 Single bits:

Bits	7	6	5	4	3	2	1	0
Octets 1	7	6	5	4	3	2	1	0

Unsigned16

An Unsigned16 is representing an unsigned number depicted by 16 bits ("enumerated"). Some application profiles (e.g. PROFIsafe) are using this data type for the coding of individual single bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
6	Unsigned16	0 ≤ i ≤ 65535	1	2 Octets

Tabelle 10- 3 Enumerated:

Bits	7	6	5	4	3	2	1	0
Octets 1	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8
Octets 2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Tabelle 10- 4 Single bits:

Bits	7	6	5	4	3	2	1	0
Octets 1	15	14	13	12	11	10	9	8
Octets2	7	6	5	4	3	2	1	0

Unsigned32

An Unsigned32 is representing an unsigned number depicted by 32 bits ("enumerated"). Some application profiles (e.g. PROFIsafe) are using this data type for the coding of individual single bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
7	Unsigned32	0 ≤ i ≤ 4294967295	1	4 Octets

Tabelle 10- 5 Enumerated:

Bits	7	6	5	4	3	2	1	0
Octets 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
Octets 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
Octets 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
Octets 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Tabelle 10- 6 Single bits:

Bits	7	6	5	4	3	2	1	0
Octets 1	31	30	29	28	27	26	25	24
Octets 2	23	22	21	20	19	18	17	16
Octets 3	15	14	13	12	11	10	9	8
Octets 4	7	6	5	4	3	2	1	0

Unsigned64

An Unsigned64 is representing a signed number depicted by 64 bits.

Numeric Identifier	Data type name	Value range	Resolution	Length
56	Unsigned64	0 ≤ i ≤ 2^{64}-1	1	8 Octets

In two's complement; the most significant bit (MSB) is the bit after the sign (SN) in the first Byte.

SN = 0: positive numbers and zero

SN = 1: negative numbers

Bits	7	6	5	4	3	2	1	0
Octets 1	2^{63}	2^{62}	2^{61}	2^{60}	2^{59}	2^{58}	2^{57}	2^{56}
Octets 2	2^{55}	2^{54}	2^{53}	2^{52}	2^{51}	2^{50}	2^{49}	2^{48}
Octets 3	2^{47}	2^{46}	2^{45}	2^{44}	2^{43}	2^{52}	2^{41}	2^{40}
Octets 4	2^{39}	2^{38}	2^{37}	2^{36}	2^{35}	2^{34}	2^{33}	2^{32}
Octets 5	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
Octets 6	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}

Bits	7	6	5	4	3	2	1	0
Octets 7	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
Octets 8	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Float32

A Float32 is representing a number defined by ANSI/IEEE 754 as single precision.

Numeric Identifier	Data type name	Value range	Resolution	Length
8	Float32	refer to ANSI/IEEE 754	refer to ANSI/IEEE 754	4 Octets

SN: sign 0 = positive, 1 = negative.

Bits	7	6	5	4	3	2	1	0
Octets 1	Exponent (E)							
	SN	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}
Octets 2	(E)	Fraction (F)						
	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
Octets 3	Fraction (F)							
	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}
Octets 4	Fraction (F)							
	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}	2^{-23}

Float64

A Float64 is representing a number defined by ANSI/IEEE 754 as single precision.

Numeric Identifier	Data type name	Value range	Resolution	Length
15	Float64	refer to ANSI/IEEE 754	refer to ANSI/IEEE 754	8 Octets

SN: sign 0 = positive, 1 = negative.

Bits	7	6	5	4	3	2	1	0
Octets 1	Exponent (E)							
	SN	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}
Octets 2	Exponent (E)				Fraction (F)			
	2^{3}	2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}
Octets 3	Fraction (F)							
	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}
Octets 4	Fraction (F)							
	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}
Octets 5	Fraction (F)							
	2^{-21}	2^{-22}	2^{-23}	2^{-24}	2^{-25}	2^{-26}	2^{-27}	2^{-28}
Octets 6	Fraction (F)							
	2^{-29}	2^{-30}	2^{-31}	2^{-32}	2^{-33}	2^{-34}	2^{-35}	2^{-36}
Octets 7	Fraction (F)							

Bits	7	6	5	4	3	2	1	0
	2^{-37}	2^{-38}	2^{-39}	2^{-40}	2^{-41}	2^{-42}	2^{-43}	2^{-44}
Octets 8	Fraction (F)							
	2^{-45}	2^{-46}	2^{-47}	2^{-48}	2^{-49}	2^{-50}	2^{-51}	2^{-52}

Visible String

This data type is defined as the ISO 646 string type. Characters are based on 8 Bit ASCII.

Numeric Identifier	Data type name	Value range	Resolution	Length
9	VisibleString	refer to ISO 646	-	variable

Bits	7	6	5	4	3	2	1	0
Octets 1	1. Character							
Octets 2	2. Character							
...	...							
Octets n	n. Character							

OctetString

An OctetString is an ordered sequence of Bytes, numbered from 1 to n.

Numeric Identifier	Data type name	Value range	Resolution	Length
10	OctetString	-	-	variable

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Character							
Octet 2	2. Character							
...	...							
Octet n	n. Character							

BYTE

A Byte is 1 octet.

Numeric Identifier	Data type name	Value range	Resolution	Length
22	Byte	-	-	1 Octet

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Byte							

WORD

A Word is an ordered sequence of 2 octets.

Numeric Identifier	Data type name	Value range	Resolution	Length
23	Word	-	-	2 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Byte							
Octet 2	2. Byte							

DWORD

A DWord is an ordered sequence of 4 octets.

Numeric Identifier	Data type name	Value range	Resolution	Length
24	DWord	-	-	4 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Byte							
Octet 2	2. Byte							
Octet 3	3. Byte							
Octet 4	4. Byte							

LWORD

A LWord is an ordered sequence of 8 octets.

Numeric Identifier	Data type name	Value range	Resolution	Length
25	LWord	-	-	8 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Byte							
Octet 2	2. Byte							
Octet 3	3. Byte							
Octet 4	4. Byte							
Octet 5	5. Byte							
Octet 6	6. Byte							
Octet 7	7. Byte							
Octet 8	8. Byte							

TimeOfDay

This data type is composed of two elements of unsigned values representing the time of day and the date. The first element is an Unsigned32 data type and contains the number of milliseconds since midnight, where midnight = 0.

The second element is of type Unsigned 16 containing the number of completed days since January 1, 1984.

Numeric Identifier	Data type name	Value range	Resolution	Length
12	TimeOfDay	$0 \le i \le (2^{28}-1)$ ms $0 \le i \le (2^{16}-1)$ days	-	6 Octets

The time is interpreted as 32 bit value. The first 4 (MSB) bits have the value zero. The date indication is coded as 16 bit value.

Bits	7	6	5	4	3	2	1	0	
Octets 1	0	0	0	0	2^{27}	2^{26}	2^{25}	2^{24}	Number of milliseconds since midnight
Octets 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octets 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octets 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octets 5	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	days since January 1st, 1984
Octets 6	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

TimeDifference

This data type is composed of two elements of unsigned values and expresses the difference in time. The first element is an Unsigned32 data type that contains the fractional portion of one day in milliseconds. The optional second element is an Unsigned16 data type that contains the difference in days.

Numeric Identifier	Data type name	Value range	Resolution	Length
13	TimeDifference	$0 \le i \le (2^{32}-1)$ ms $0 \le i \le (2^{16}-1)$ days	-	4 or 6 Octets

The time is interpreted as 32 bit value. The first 4 (MSB) bits have the value zero. The date indication is coded as 16 bit value.

Bits	7	6	5	4	3	2	1	0	
Octets 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Number of milliseconds (of one day)
Octets 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octets 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octets 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octets 5	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	Number of days (optional)
Octets 6	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

Date

This data type is composed of six elements of unsigned values and expresses calendar date and time. The first element is an Unsigned16 data type and gives the fraction of a minute in milliseconds. The second element is an Unsigned8 data type and gives the fraction of an hour in minutes. The third element is an Unsigned8 data type and gives the fraction of a day in hours with the most significant bit indicating Standard Time or Daylight Saving Time. The fourth element is an Unsigned8 data type. Its upper three (3) bits give the day of the week and its lower five (5) bits give the day of the month. The fifth element is an Unsigned8 data type and gives the month. The last element is Unsigned8 data type and gives the year. The values 0 ... 50 correspond to the years 2000 to 2050; the values 51 ... 99 correspond to the years 1951 to 1999.

Numeric Identifier	Data type name	Value range	Resolution	Length
50	Date	0 ms ≤ i ≤ 99 years	-	7 Octets (Unsigned16 + 5 x Unsigned8)

Bits	7	6	5	4	3	2	1	0	
Octet 1	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	0...59999 milliseconds
Octet 2	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octet 3	res	res	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	0...59 minutes
Octet 4	SU	res	res	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	0...23 hours
Octet 5	day of week			day of month					1...7 day of week
	2^{2}	2^{1}	2^{0}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	1...31 day of month
Octet 6	res	res	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	1...12 month
Octet 7	res	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	0...99 years

SU: Standard Time = 0; Daylight Saving Time = 1

day of week: 0 Monday = 1; Tuesday =2.....Sunday = 7

res = reserved

TimeOfDay without date indication

This data type consists of one element of an unsigned value and expresses the time of day without date indication. The element is an Unsigned32 data type and contains the time after midnight in milliseconds.

Numeric Identifier	Data type name	Value range	Resolution	Length
52	TimeOfDay without date indication	0 ≤ i ≤ (2^{28}-1)	ms	4 Octets (Unsigned32)

The time is interpreted as 32 bit value.

Bits	7	6	5	4	3	2	1	0	
Octet 1	0	0	0	0	2^{27}	2^{26}	2^{25}	2^{24}	Number of milliseconds since midnight
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

TimeDifference with Date indication

This data type is composed of two elements of unsigned values that express the difference in time. The first element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds. The second element is an Unsigned16 data type that provides the difference in number of days.

Numeric Identifier	Data type name	Value range	Resolution	Length
53	TimeDifference with date indication	$0 \le i \le (2^{32}-1)$ ms $0 \le i \le (2^{16}-1)$ days	ms, days	6 Octets (Unsigned32 + Unsigned16)

The time is interpreted as 32 bit value. The date indication is coded as 16 bit value.

Bits	7	6	5	4	3	2	1	0	
Octet 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Number of milliseconds (of one day)
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octet 5	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	Number of days
Octet 6	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

TimeDifference without Date indication

This data type consists of one element of an unsigned value that expresses the difference in time. he element is an Unsigned32 data type providing the fractional portion of one day in milliseconds.

Numeric Identifier	Data type name	Value range	Resolution	Length
54	TimeDifference without date indication	$0 \le i \le (2^{32}-1)$ ms	ms, days	4 Octets (Unsigned32)

The time is interpreted as 32 bit value.

Bits	7	6	5	4	3	2	1	0	
Octet 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Number of milliseconds (of one day)
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

NetworkTime

This data type is based on the RFC 1305 standard and composed of two unsigned values that express the network time related to a particular date. Its semantic has changed in IEC 61158-6:2003.

The first element is an Unsigned32 data type that provides the network time in seconds since 1.1.1900 0.00,00(UTC) or since 7.2.2036 6.28,16(UTC) for time values less than 0x9DFF4400, which represents the 1.1.1984 0:00,00(UTC). The second element is an Unsigned32 data type that provides the fractional portion of seconds in $1/2^{32}$ s. Rollovers after 136 years are not automatically detectable and are to be maintained by the application.

Numeric Identifier	Data type name	Value range	Resolution	Length
58	NetworkTime	Byte 1 to 4: 0 ≤ i ≤ (232-1) Byte 5 to 8: 0 ≤ i ≤ (232-1)	S $(1/2^{32})$ s	8 Octets (Unsigned32 + Unsigned32)

Bits	7	6	5	4	3	2	1	0	
Octet 1	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Number of seconds since 1.1.1900. Rollover after 136 years. Thus, next would be 7.2.2036.
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octet 5	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Fractional portion of seconds: $1/2^{32}$ s
Octet 6	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 7	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 8	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

The Time Stamp mechanism in PROFIBUS DP is using this data type. However, the semantic is slightly different:

- The least significant Bit of the fractional portion (2^0) is device internally used to indicate a synchronized or unsynchronized state of the clock time.

NetworkTimeDifference

This data type is composed of an integer value and of an unsigned value that express the difference in network time. The first element is an Integer32 data type that provides the network time difference in seconds. The second element is an Unsigned32 data type that provides the fractional portion of seconds in $1/2^{32}$ s.

Numeric Identifier	Data type name	Value range	Resolution	Length
59	NetworkTimeDifference	Byte 1 to 4: $-2^{31} \le i \le (2^{31}-1)$ Byte 5 to 8: $0 \le i \le (2^{32}-1)$	S $(1/2^{32})$ s	8 Octets (Integer32 + Unsigned32)

Bits	7	6	5	4	3	2	1	0	
Octet 1	SN	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Signed number of seconds
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
Octet 5	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	Fractional portion of seconds: $1/2^{32}$ s
Octet 6	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 7	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 8	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

Siehe auch

PROFIdrive-spezifische Datentypen (Seite 277)

10.2 Profilspezifische Datentypen PROFIBUS/PROFINET (nur in englisch verfügbar)

Existing PROFIBUS/PROFINET profile specific data types

This topic contains the profile specific data types of PROFIBUS/PROFINET.

Float32+Unsigned8 (former "DS33")

This data structure consists of the value and the status of a Float32 parameter. The parameter can be an input or output..

Numeric Identifier	Data type name	Value range	Resolution	Length
101	Float32 +Unsigned8	See Float32 and Unsigned8	-	5 Octets

SN: sign 0 = positive, 1 = negative

Bits	7	6	5	4	3	2	1	0
Octets 1	Exponent (E)							
	SN	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}
Octets 2	(E)	Fraction (F)						
	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
Octets 3	Fraction (F)							
	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}
Octets 4	Fraction (F)							
	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}	2^{-23}
Octet 5	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Unsigned8+Unsigned8 (former "DS34")

This data structure consists of the value and the status of the Unsigned8 parameter. The parameter can be an input or output.

Numeric Identifier	Data type name	Value range	Resolution	Length
102	Unsigned8 + Unsigned8	See Unsigned8	-	2 Octets

In two's complement; the most significant bit (MSB) is the bit after the sign (SN) in the first Byte.

Bits	7	6	5	4	3	2	1	0
Octet 1	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Octet 2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

OctetString2+Unsigned8 (former "DS35")

This data structure consists of the value and the status of the OctetString parameter. The parameter can be input or output.

Numeric Identifier	Data type name	Value range	Resolution	Length
103	OctetString2 + Unsigned8	see OctetString2 and Unsigned8	-	3 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	1. Byte							
Octet 2	2. Byte							
Octet 3	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Unsigned16_S

This data structure consists of the value and the status embedded in an unsigned 16 data type. The parameter using this data type can be input or output.

Numeric Identifier	Data type name	Value range	Resolution	Length
104	Unsigned16_S	0 to 2^{13} + 0 to 3	1	2 Octets

Bits	7	6	5	4	3	2	1	0
Octets 1	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6
Octets 2	2^5	2^4	2^3	2^2	2^1	2^0	St1	St0

st1 (bit1)	st0 (bit0)	Meaning
0	0	input channel: bad (value is fail-safe value) output channel: reserved
0	1	input channel: simulation output channel: reserved
1	0	input channel: uncertain output channel: reserved
1	1	input channel: good output channel: reserved

Integer16_S

This data structure consists of the value and the status embedded in an integer 16 data type. The parameter using this data type can be input or output.

Numeric Identifier	Data type name	Value range	Resolution	Length
105	Integer16_S	- 2^{12} to 2^{12} –1 + 0 to 3	1	2 Octets

SN: sign 0 = positive, 1 = negative

Bits	7	6	5	4	3	2	1	0
Octets 1	SN	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}
Octets 2	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	St1	St0

st1 (bit1)	st0 (bit0)	Meaning
0	0	input channel: bad (value is fail-safe value) output channel: reserved
0	1	input channel: simulation output channel: reserved
1	0	input channel: uncertain output channel: reserved
1	1	input channel: good output channel: reserved

Unsigned8_S

This data structure consists of the value and the status embedded in an Unsigned8 data type. The parameter using this data type can be an input or output.

Numeric Identifier	Data type name	Value range	Resolution	Length
106	Unsigned8_S	0 to 2^{5} + 0 to 3	1	1 Octet

Bits	7	6	5	4	3	2	1	0
Octets 1	2^5	2^4	2^3	2^2	2^1	2^0	St1	St0

st1 (bit1)	st0 (bit0)	Meaning
0	0	input channel: bad (value is fail-safe value) output channel: reserved
0	1	input channel: simulation output channel: reserved
1	0	input channel: uncertain output channel: reserved
1	1	input channel: good output channel: reserved

OctetString_S

This data structure consists of the value and the status embedded in an octet string data type. The parameter using this data type can be an input or output.

In the scope of this profile, this is the preferred data type for digital values. It contains the value of a binary channel as a bit in a value field (ch(x)), and a status information in a status field (st1(x) and st0(x)) coded as shown below. If a channel is unused, its position in the value field and in the status field has to be set to 0.

The value field and the status field are packed into an OctetString with the bit coding as shown below. This data type is defined as OctetString_S.

Numeric Identifier	Data type name	Value range	Resolution	Length
107	OctetString_S	See OctetString and below	-	n Octets

Bits	bit7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Octet 1	ch(8)	ch(7)	ch(6)	ch(5)	ch(4)	ch(3)	ch(2)	ch(1)
***	***	***	***	***	***	***	***	***
Octet m	ch(n)	ch(n-1)	ch(n-2)	ch(n-3)	ch(n-4)	ch(n-5)	ch(n-6)	ch(n-7)
Octet m+1	st1(4)	st0(4)	st1(3)	st0(3)	st1(2)	st0(2)	st1(1)	st0(1)
***	***	***	***	***	***	***	***	***
Octet 3*m	st1(n)	st0(n)	st1(n-1)	st0(n-1)	st1(n-2)	st0(n-2)	st1(n-3)	st0(n-3)

ch(x) value for channel x; st(x) status information for channel x; $1<x\leq n$

st1 (bit1)	st0 (bit0)	Meaning
0	0	input channel: bad (value is fail-safe value) output channel: reserved
0	1	input channel: simulation output channel: reserved
1	0	input channel: uncertain output channel: reserved
1	1	input channel: good output channel: reserved

F message trailer with 4 octets

This data structure consists of the status/control byte, consecutive number, and 2 byte CRC parameters in PROFIsafe's V1-mode or status/control byte and 3 byte CRC parameters in PROFIsafe's V2- mode. This data type can be associated with input or output data up to 12 byte. So far the data type "OctetString" with numeric identifier "10" has been used for the coding of the F message trailer. For new product developments the new data type specification shall be considered.

Numeric Identifier	Data type name	Value range	Resolution	Length
110	F message trailer with 4 octets	PROFIsafe: V1-mode and V2-mode	-	4 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	Status/Control octet							
Octet 2	Consecutive number (V1-mode) or High-octet CRC(V2-mode) *)							
Octet 3	CRC *)							
Octet 4	Low-byte CRC (least significant octet) *)							

* Byte ordering according to IEC 61158-5 Type 3

F message trailer with 5 octets

This data structure consists of the status/control byte and 4 byte CRC parameters in PROFIsafe's V2-mode. This data type can be associated with input or output data up to 122 byte.

Numeric Identifier	Data type name	Value range	Resolution	Length
111	F message trailer with 5 octets	PROFIsafe: V2-mode	-	5 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	Status/Control octet							
Octet 2	1. byte CRC (most significant octet) *)							
Octet 3	2. byte CRC *)							
Octet 4	3. byte CRC *)							
Octet 5	4. byte CRC (least significant octet) *)							

* Byte ordering according to IEC 61158-5 Type 3

F message trailer with 6 octets

This data structure consists of the status/control byte, consecutive number and 4 byte CRC parameters in PROFIsafe's V1-mode. This data type can be associated with input or output data up to 122 byte. So far the data type "Octet String" with numeric identifier "10" has been used for the coding of the F message trailer. For new product developments the new data type specification shall be considered.

Numeric Identifier	Data type name	Value range	Resolution	Length
111	F message trailer with 6 octets	PROFIsafe: V1-mode	-	6 Octets

Bits	7	6	5	4	3	2	1	0
Octet 1	Status/Control octet							
Octet 2	Consecutive number							
Octet 3	1. byte CRC (most significant octet) *)							
Octet 4	2. byte CRC *)							
Octet 5	3. byte CRC *)							
Octet 6	4. byte CRC (least significant octet) *)							

* Byte ordering according to IEC 61158-5 Type 3

Siehe auch

PROFIdrive-spezifische Datentypen (Seite 277)

Index

K

L

P

R

S

T

U

X