Konzepte für die Kommunikation zwischen Automatisierungsgeräten

von Diplom-Ingenieur
Qimin Zhang

von der Fakultät IV – Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr. P. Pepper
Gutachter: Prof. Dr.-Ing. D. Naunin
Gutachter: Prof. Dr.-Ing. G. Hommel

Tag der wissenschaftliche Aussprache: 3. September 2002

Berlin 2002
D 83

Vorwort

Die vorliegende Arbeit entstand während meine Tätigkeit als Stipendiat bei der Siemens AG, Bereich A&D, AS E in Karlsruhe für ein gemeinsames Projekt zwischen dem Institut für Elektronik und Lichttechnik der Technischen Universität Berlin und der Siemens AG.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. D. Naunin, dem Inhaber des Lehrstuhls für Elektronik und Lichttechnik im Fachbereich Elektrotechnik an der Technische Universität Berlin, der die Arbeit immer richtungsweisend und kritisch begleitete und sie durch Vorschläge und Verbesserungen förderte.

Herrn Professor Dr.-Ing. G. Hommel danke ich für die Übernahme des Korreferats, die Durchsicht des Manuskripts der Arbeit und die wertvollen Vorschläge.

Mein herzlicher Dank gilt Herrn Dr.-Ing. U. Zahner, Siemens AG, der mich in zahlreichen Diskussionen unterstützt und durch seinen reichhaltigen Fundus an praktischer Erfahrung dieser Arbeit immer wieder anregende Impulse verliehen hat.

Ebenfalls möchte ich der Siemens AG meinen Dank aussprechen. Sie förderte die Arbeit durch finanzielle Unterstützung sowie durch gute Arbeitsbedingung.

Schließlich danke ich meinem Vater Shouyi Zhang und meiner Mutter Youchen Lin, die großes Verständnis haben und mir bei der Arbeit stets unterstützen.

Inhaltsverzeichnis

Liste der Abkürzungen

Liste der Abkürzungen

AG	: Automatisierungsgerät
AS	: Automatisierungssystem
ASI	: Aktor-Sensor-Interface
AWL	: Anweisungsliste
BK	: Blockorientierte Kommunikation
BLS	: Betriebsleitsystem
BRCV	: Blockorientiertes Empfangen
BSEND	: Blockorientiertes Senden
BuB	: Bedienen und Beobachten
CAE	: Computer Aided Engineering
CFB	: Kommunikationsfunktionsbaustein
CFC	: Continuous Function Chart
CP	: Communication Processor, Kommunikationsprozessor
CPU	: Central Processor Unit, Zentralbaugruppe
CR	: Central Rack, Zentralbaugruppenträger
DB	: Datenbaustein
DIN	: Deutsche Industrie Norm
DOCPRO	: Dokumentationswerkzeug
DP	: Dezentrale Peripherie
EN	: Europäische Norm
ER	: Extension Rack, Erweiterungsbaugruppenträger
ES	: Engineeringssystem
FB	: Funktionsbaustein
FC	: Function
FDL	: Fieldbus Data Link Layer
FM	: Funktionsmodul
FMS	: Fieldbus Message Specification
FUP	: Funktionsplan
HMI	: Human Machine Interface
IBS	: Inbetriebsetzung
IEEE	: The Institute of Electrical and Electronics Engineers, USA

ID	: Identifikationsnummer
IK	: Implizite Kommunikation
IM	: Interface Module, Anschaltungsbaugruppen
ISO	: International Standards Organization
I/O	: Input/Output
K-Bus	: Kommunikationsbus
KOP	: Kontaktplan
LWL	: Lichtwellenleiter
MPI	: Multi Point Interface
OB	: Organisationsbaustein
OP	: Operator Panel
OS	: Operator System
OSI	: Open System Interconnection
P-Bus	: Peripheriebus
PG	: Programmiergerät
PNK	: Prozeßnahe Komponente
PS	: Power Supply
SCL	: Structured Control Language
SDB	: System-Datenbaustein
SFB	: System-Funktionsbaustein
SFC	: Systemfunktion
SFC	: Sequential Function Chart
SM	: Signal Module, Signalbaugruppen
SPS	: Speicherprogrammierbare Steuerung
UR	: Universal Rack
URCV	: Unkoordiniertes Empfangen
USEND	: Unkoordiniertes Senden
WinCC	: Windows Control Center

1 Einleitung und Aufgabenbeschreibung

In der Leittechnik hat man sich zunehmend damit auseinanderzusetzen, daß die Automatisierungssysteme immer größer und komplexer werden. Es ist in einer großen technischen Anlage sowohl technisch als auch wirtschaftlich nicht mehr zufriedenstellend, alle Komponente der Anlage zentral zu verwalten. Die Entwicklung leistungsfähigerer Mikroprozessoren hat es inzwischen ermöglicht, komplexe Aufgaben nicht mehr von einem zentralen Rechner, sondern vor Ort in der Steuerung erledigen zu lassen. Dies führt zu einer Dezentralisierung von Automatisierungslösungen. Der Trend zur dezentralen Schaltungstechnik wird sich in Zukunft fortsetzen, denn, wie sich in der Datenverarbeitung und in der Energieversorgung bereits gezeigt hat, dezentrale Konzepte haben beispielsweise den Vorteil, daß Veränderungen im Prozeßsystem auch bei genau zugeschnittenen Automatisierungslösungen leichter realisiert werden können. Ein anderer leicht einsehbarer Grund hierfür: Durch das Auslagern von Funktionen vor Ort – dorthin, wo sie auch wirklich gebraucht werden – ergibt sich durch den geringer werdenden Installations- und Verdrahtungsaufwand eine deutliche Kostenersparnis. Auf der anderen Seite wird hierdurch eine Entlastung der zentralen Komponenten erreicht. Die Konsequenz ist deshalb langfristig die Notwendigkeit prozeßnaher und dezentraler Kommunikation ohne Beteiligung zentraler Komponenten.

Eine Dezentralisierung bedeutet zugleich eine hierarchische, funktionsorientierte Strukturierung, die nur noch eingeschränkt auf gerätespezifische Rahmenbedingungen Rücksicht nehmen kann: Zum einen müssen wegen der räumlichen Verteilung oder der großen Mengengerüste Automatisierungsaufgaben z.T. auf mehrere Automatisierungsgeräte verteilt werden, zum anderen werden für eine hohe Auslastung der teuren Investitionsgüter mehrere Automatisierungsaufgaben auf einem System implementiert. Die Folge ist, daß moderne Engineering-Werkzeuge geräteübergreifend agieren müssen. Dabei spielt die Kommunikation zwischen den Automatisierungsgeräten eine wesentliche Rolle. Durch die dezentrale Struktur vermehren sich zwangsweise die Kommunikationsaufgaben und erhöht sich damit verbundener Projektierungsaufwand. Der Schlüssel für eine Dezentralisierung ist somit ein Kommunikationsnetzwerk, das es ohne größeren Projektierungsaufwand ermöglicht, Informationen für alle im Automatisierungsverbund integrierten Komponenten direkt zur Verfügung zu stellen. Das heißt ein transparenter Datenaustausch findet statt, sowohl horizontal durch den Produktionsbereich als auch vertikal durch die gesamte Fabrik.

Die heutige Projektierung von Automatisierungslösungen erfolgt überwiegend gerätebezogen. Pro Automatisierungssystem werden die gewünschten Funktionen projektiert. Es verbleibt ein Anteil an übergreifender Funktionalität, für die eine Kommunikation zwischen Automatisierungssystemen explizit projektiert werden muß. Der technische Stand für die Kommunikationsprojektierung basiert zur Zeit überall auf dem Bausteinkonzept /PCS97/. Von dem Hersteller werden die vorgefertigten Bausteine angeboten. Mit dessen Hilfe kann der Benutzer dann die geräteübergreifende Kommunikation einrichten. Das generelle Vorgehen bei der Projektierung von Kommunikation zwischen Automatisierungsgeräten besteht aus drei Schritten,

- Hardware,

- Netzprojektierung,
- Technologische Funktionen.

An der **Hardware** wird die Zuordnung zu den Subnetzen festgelegt und die Teilnehmeradresse in den Subnetzen vergeben. Die **Netzprojektierung** übernimmt die Aufgabe für die Festlegung der Netztopologie, der Bussysteme (z.B. Industrial Ethernet, Profibus, MPI) und der Verbindungswege zwischen den Teilnehmern am Anlagenbus. Die **technologischen Funktionen** werden mit Hilfe von CFC- und SFC-Plänen (Continuous und Sequential Function Chart-Plänen) projektiert. Funktionsbausteine sind atomare Objekte und damit unteilbar einer CPU zugeordnet.

Während die Funktionsbausteine innerhalb einer CPU ihre Signale gegenseitig auf einfache Art, nämlich durch gezielten Speicherzugriff, austauschen können, müssen bei übergreifender Funktionalität die Kommunikationsbausteine für den notwendigen Datenaustausch zwischen Automatisierungssystemen extra projektiert werden. Im Bezug auf immer größer werdenden Aufwand für die Kommunikationsprojektierung sollte ein modernes Kommunikationskonzept das Projektierungs-Werkzeug (CFC) in Zukunft erlauben, die Projektierungsaufgabe implizit ohne Zutun des Anwenders zu übernehmen. Außerdem machen die Nachteile mit bisheriger Methode es auch immer notwendiger, ein neues Kommunikationskonzept für SIMATIC zu entwerfen, das im Vergleich zum technischen Stand die aktuellen Probleme löst oder zumindest mindert. Die Hauptprobleme werden im folgenden kurz erläutert.

Heute muß, um das Signal an den richtigen Baustein zu senden, eine eindeutige Adresse parametriert werden. Dieses kann am Eingang eines spezifischen Kommunikationsbausteins in expliziter Form geschehen (Abbildung 1.1).

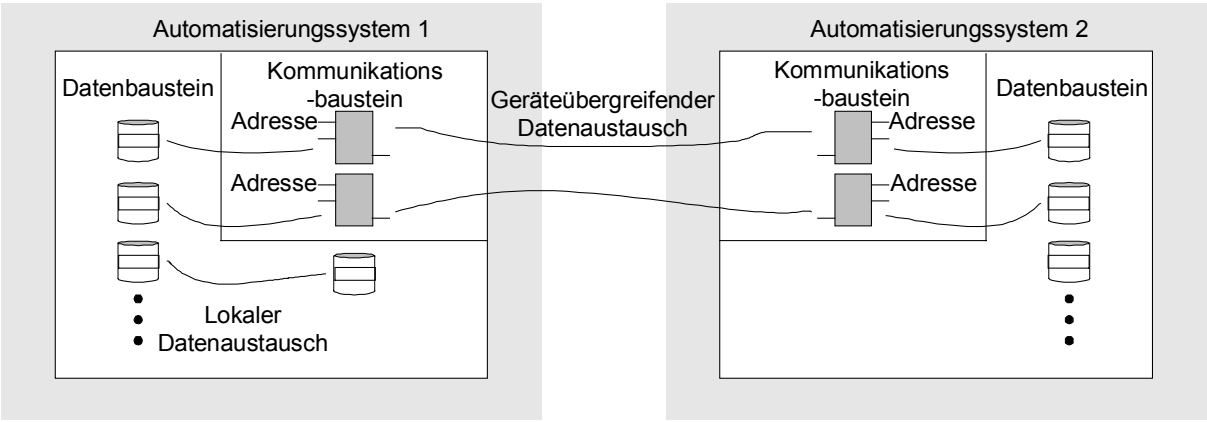


Abbildung 1.1: Übergreifende Funktionalität

Weil Ziel und Quelle jedesmal unterschiedlich sind, muß jede Verbindung, die als Kommunikationskanal zwischen Automatisierungsfunktionen in unterschiedlichen Automatisierungsgeräten fungiert, separat in CFC-Plänen projektiert werden. Bei möglichen künftigen Änderungen, die einen Kommunikationspartner betreffen, z.B. Verschiebung eines

Kommunikationspartners von einem Automatisierungssystem auf ein anderes, muß die Verbindung komplett neu projektiert werden. Das heißt unter anderem, neue Kommunikationsbausteine einzufügen und entsprechend zu parametrieren. Bei einer Funktionsverschiebung beispielsweise von über hundert Signalen ist der Projektierungsaufwand enorm. Der Folgeeffekt ist deshalb, daß eine weitere Benutzung von vorher projektierten Funktionsplänen für eine andere Anwendung schwierig erscheint.

Um die Erstellung von Plänen, die eine Fülle von Automatisierungsfunktionen enthalten (u.U. >1000), von den gerätebezogenen Rahmenbedingungen abkoppeln zu können (Lokalisation, Leistung, Anzahl der I/Os,...), besteht der Wunsch, einen Gesamtplan zwanglos auf mehrere Automatisierungssysteme verteilen zu können. Dies hat zur Folge, daß einfache Speicherzugriffe innerhalb der Steuerung jetzt u.U. über die Gerätegrenzen hinaus gehen und eine Kommunikation zwischen den Steuerungen erzwingen. Hierbei muß vermieden werden, daß die Kommunikationsmechanismen applikativ neu projektiert werden müssen. Es bedarf Mechanismen der impliziten Kommunikation, die den Zugriff auf externe Größen verdecken.

Bisher läuft die Datenübertragung zyklisch, und die Zykluszeit sowie deren Priorität hängt davon ab, in welchem Organisationsbaustein (OB) die Kommunikationsmechanismen eingebaut sind. In der Praxis erwies sich die Zuteilung der jeweils richtigen Zykluszeit als schwierig. Mit einer zu langsam eingestellten Zykluszeit kann die Systemdynamik nicht richtig repräsentiert werden (Aliasing Effekt). Das führt zu einer falschen Einschätzung und somit zu einer möglichen Fehlreaktion von Aktoren. Aber man kann auch nicht in allen Fällen eine schnelle Zykluszeit einsetzen. Da die Daten hier in kleinen Zeitabständen zyklisch übertragen werden, entsteht eine hohe Busbelastung. Dies ist unabhängig davon, ob das Ziel einen neuen Wert wünscht oder nicht. In diesem Fall kann es vorkommen, daß durch eine hohe Busbelastung die Anforderungen von anderen Busteilnehmern nicht rechtzeitig erfüllt werden können. Außerdem verursacht die schnelle zyklische Bearbeitung für den Datenaustausch eine zusätzliche Rechenlast im jeweiligen Automatisierungssystem. Diese Problematik soll bei dem Entwurf berücksichtigt werden. Das neue Konzept soll ein Automatisierungssystem in die Lage versetzen, daß es den Bus möglichst wenig belastet und gleichzeitig die Übertragung von notwendigen Daten nicht vernachlässigt.

Das Ziel ist, daß die Datenübertragung abhängig von der Dynamik des Prozesses in einem variablen Taktraster stattfinden kann. Dem jeweiligen Signal werden verschiedene Kommunikationsdienste mit unterschiedlichem Übertragungszyklus angeboten. Man kann einen „langsamen" und „schnellen" Dienst einführen, dessen Zykluszeit sich jeweils an der praktischen Anwendung orientiert. Für die Sicherheitsgüte der Übertragung soll es auch eine Alternative geben, den „gesicherten“ bzw. „ungesicherten“ Dienst. Bei dem Konzept soll außerdem noch auf die Granularität der zu übertragenden Signale geachtet werden. Es handelt sich hier um verschiedene Datenanzahl und Datentypen. Es soll angestrebt werden, beliebig viele Datentypen in variabler Anzahl über einen Systemmechanismus übertragen zu können.

Zur Unterstützung der arbeitsteiligen Projektierung und der schrittweisen Inbetriebnahme muß weiterhin ein entkoppeltes Arbeiten an verschiedenen Automatisierungssystemen möglich sein. Das heißt beispielsweise, daß ein Automatisierungssystem möglichst vollständig generiert und in Betrieb genommen werden kann, auch ohne daß die

Kommunikationsbeziehungen schon generierbar sind, weil etwa eine Partnerressource fehlt. Desweiteren muß die Generierung der impliziten Kommunikation (IK) möglichst rückwirkungsfrei auf die Generierung des eigentlichen Programms auf dem Automatisierungssystem sein, damit nicht nach IK-Generierung noch einmal eine vollständige Inbetriebnahme notwendig wird.

Zur Lösung der aufgezeigten Problemen soll diese Arbeit einen Beitrag leisten. Die Gliederung der Arbeit ist wie folgt:

Kapitel 2 und Kapitel 3 erklären die prinzipielle Arbeitsweise und spezifische technische Eigenschaften einer speicherprogrammierbaren Steuerung (SPS). Es enthält eine Einführung in das Automatisierungspaket SIMATIC des Siemens AG. Danach werden die Komponenten wie Schichtenmodell, Kommunikationsfunktion, Netztypen, usw. genauer beschrieben. Zusammen bilden sie das kommunikationstechnische Umfeld industrieller Kommunikation.

Im Kapitel 4 werden die künftigen Anforderungen industrieller Kommunikation in der Zellenebene analysiert. Ein Sitzungsmodell wird angegeben, an dem alle Konzepte sich orientieren sollen, um eine künftige Integration zu gewährleisten. Am Ende werden die Meßgrößen bezüglich der Performanceanalyse angegeben.

Nach den technischen Anforderungen in Kapitel 4 werden Lösungsansätze und mögliche Realisierungen in den Kapiteln 5 bis 7 untersucht. Drei Kommunikationsvarianten, die explizite Methode, die blockorientierte Übertragung und die implizite Kommunikation, werden schwerpunktmäßig analysiert, wobei die explizite Methode auch teilweise den aktuellen technischen Stand wiederspiegelt. Am Ende wird eine Performancemessung durchgeführt.

Vor- und Nachteile des jeweiligen Konzeptes werden in Kapitel 8 verglichen. Kapitel 9 faßt die Ergebnisse der Arbeit zusammen. In einem Ausblick wird auf Möglichkeiten der weiteren Entwicklung hingewiesen.

2 Industrielle Steuerungssysteme und Testanordnung

In allen Bereichen der Automatisierungstechnik findet man Prozeßleitsysteme, die die Funktionalitäten der Steuerung mit der Visualisierung des technischen Prozesses vereinen /Pol94/, /VDE90/. Diese Zweiteilung hat sich wegen der unterschiedlichen Anforderungen und Medien herauskristallisiert. In der Nähe des Prozesses bestehen Echtzeitanforderungen und zwingen zum Einsatz von echtzeitfähigen Systemen. In einer zentralen Kontrollwarte können eher Standardsysteme zum Einsatz kommen, die auch in der Office-Welt verwendet werden. Vermehrt fallen aber auch hier die Defizite der nicht echtzeitfähigen, weniger stabilen Betriebssysteme ins Gewicht, da zentrale Aufgaben (Server, Sicherheitsapplikationen, Archive) ebenfalls dort bearbeitet werden. Es kommt deshalb wieder der Wunsch nach besser geeigneten Systemen für diese Aufgaben auf.

2.1 Funktionsebenen

In der Automatisierungslandschaft kann die Steuerungskette hierarchisch in fünf Automatisierungsebenen unterteilt werden, die jeweils spezifische Aufgabe haben (Abbildung 2.1) /Ben90/.

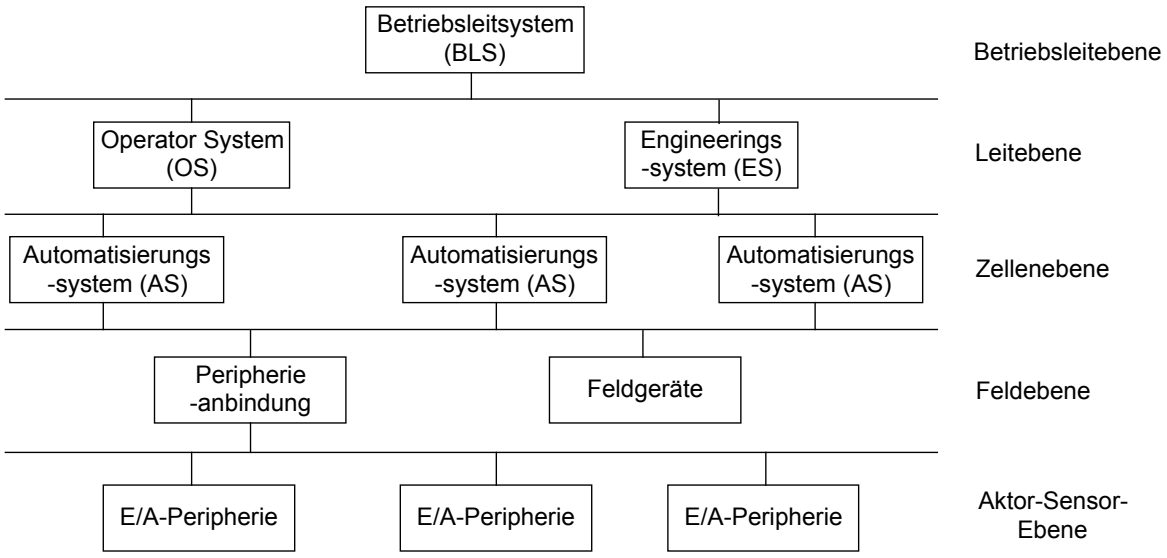


Abbildung 2.1: Automatisierungsebenen

Betriebsleitebene

In der Betriebsleitebene werden übergeordnete Aufgaben bearbeitet, die die zentrale Leitung des ganzen Automatisierungssystems betreffen (Managementfunktionen) /Rem79/.

Ein Betriebsleitsystem wird für die folgenden Aufgaben verwendet:

- Gewährleistung des ökonomischen und sicheren Betriebs der Anlage,
- Steuerung der untergeordneten Ebenen,
- Erarbeitung von alternativen Strategien unter geänderten Betriebsbedingungen oder beim Ausfall von Prozessstationen.

Ein Terminalbus verbindet das gesamte Betriebsleitsystem mit der Welt der Datenverarbeitung. Im Gegensatz zum zeitkritischen Systembus stehen hier Echtzeitfähigkeit und Redundanz nicht im Vordergrund. Es wird meistens ein Protokoll der Büroanwendung, z.B. Ethernet und TCP/IP, benutzt, so daß eine homogene Kommunikation mit anderen EDV-Geräten stattfinden kann.

Leitebene

Für die Führung eines Prozesses ist es notwendig, den aktuellen Prozeßzustand zu beobachten und gegebenenfalls einzugreifen. Der Anwender beobachtet den Prozeß durch das Operator System (OS). Es zeigt die aktuelle Prozeßabbildung.

Die Aufgaben des Rechners auf dieser Ebene sind wie folgt:

- Speichern von Prozeßwerten,
- analysierende und optimierende Verarbeitungsfunktionen,
- Ausgabe der Verarbeitungsfunktionen in Protokollform.

Die dafür erforderlichen Daten werden standortübergreifend gesammelt und verarbeitet. Ebenso kann von der Leitebene aus auf andere Standorte zugegriffen werden. Die Anzahl der Teilnehmer kann mehr als 1.000 betragen.

Die Funktionalität sowohl in einer SPS als auch in einem OS muß vorher projektiert werden. Das geschieht durch das ES. Es ermöglicht auch die Konfiguration der Systemfunktionalität.

Beide Anforderungen (Prozeßführung und Engineering) können nur auf der Basis robuster bzw. gut verfügbarer Systeme befriedigend erfüllt werden. Die Redundanz von Daten und Systemen ist deshalb eine wichtige Maßnahme, um den Ausfall von Systemen und den Verlust von Daten bewältigen zu können. Hierfür muß auf der Basis gewöhnlicher Betriebssysteme in der Regel ein derartiger Aufwand getrieben werden, daß hier vermehrt der Wunsch nach speziellen Lösungen entstanden ist (Unix, ...)

Zellenebene

Die Zellenebene enthält die Automatisierungssysteme (AS). Jedes AS hat mindestens eine CPU, auf der die Steuerungsaufgabe läuft.

Zu den Aufgaben der Zellenebene gehören:

- Abarbeiten der Regelungs-, Steuerungs- bzw. Optimierungsaufgaben,
- Einlesen von Istwerten und Ausgabe von Sollwerten,
- Überprüfung der Regler und Messgeräte,

- Alarm oder Meldung von Betriebsstörungen an die Leitebene.

Automatisierungssysteme werden durch einen Systembus untereinander und mit der übergeordneten Ebene verbunden, an dem sich auch weitere Komponenten anschließen können. In Betrieb tauschen Automatisierungssysteme die Daten unter sich aus. Um den Anspruch an eine hochpriore Datenübertragung zu erfüllen, werden hier spezielle Protokolle verwendet.

Feldebene

Die Feldebene ist das Bindeglied zwischen den Prozeßanlagen und den Automatisierungsgeräten. Die Feldgeräte messen, melden und geben die Befehle der Zellenebene an die Anlagen weiter. Es werden überwiegend kleine Datenmengen übertragen. Typisch ist eine hierarchische Kommunikation, d.h. mehrere Feldgeräte kommunizieren mit einem Master.

Die Feldsignale vom Prozeß werden dem Automatisierungssystem zugeführt. Für die Signalumwandlung ist jeweils eine Ein- bzw. Ausgabeeinheit zuständig. Außerdem besteht die Möglichkeit, einen konkreten Sensor oder Aktor direkt an den Bus anzuschließen, so daß hierfür I/O-Baugruppen entfallen können.

Aktor-Sensor-Ebene

In dieser Ebene kommuniziert ein Master mit den an seinem Subnetz angeschlossenen Aktoren und Sensoren.

Typische Aufgaben sind:

- Eingabe (analog oder digital),
- Ausgabe (analog oder digital),
- Zählereingabe,
- Frequenzeingabe,
- Impulsausgabe.

Die I/O-Komponenten sind durch Aktor-/Sensor-Interface (A/S-Interface) an die Peripherieanschaltung angeschlossen, die wiederum zum Automatisierungssystem führt. Kennzeichen für die Datenübertragung sind schnelle Reaktionszeiten für wenige Datenbits.

2.2 Funktionsweise des Automatisierungssystems

In der konventionellen Steuerungstechnik kann eine Steuerungsaufgabe durch Verdrahtung von Schützen bzw. Relais sowie Schaltuhren bzw. Verzögerern individuell gelöst werden. Je nach der Aufgabenstellung ist die Konfiguration der Hardware unterschiedlich. Man spricht deshalb bei Schützen- und Relaissteuerungen sowie bei elektronischen Steuerungen, die aus einzelnen Baugruppen zusammengebaut werden, von verbindungsprogrammierten

Steuerungen. Dank moderner Komponenten der Halbleiterbranche wird diese aufwendige und kostenintensive Methode durch ein digitales Automatisierungssystem ersetzt, bei dem die Steuerungsaufgabe in Form eines Anwenderprogramms in der CPU hinterlegt und abgearbeitet wird.

Es gibt viele Sorten von Automatisierungssystemen. Deren funktionale Prinzipien bleiben aber immer gleich. Die wichtigsten Merkmale einer speicherprogrammierbaren Steuerung (SPS) werden in Abschnitt 2.2.1 und 2.2.2 erläutert.

2.2.1 Organisation der Programmausführung

Das Gesamtprogramm einer Zentralbaugruppe (CPU) besteht aus dem Betriebssystem und dem Anwenderprogramm. Das Betriebssystem ist die Gesamtheit aller Anweisungen und Vereinbarungen geräteinterner Betriebsfunktionen. Das Anwenderprogramm ist die Gesamtheit aller Anwenderdaten, nämlich programmierte Anweisungen und Vereinbarungen für die Signalverarbeitung, durch die eine zu steuernde Anlage bzw. der zu steuernde Prozeß gemäß der Steuerungsaufgabe beeinflußt wird. Es nutzt die Dienste des Betriebssystems, im besonderen die des Ablauftimings und der I/O-Baugruppen- bzw. Busansteuerung, d.h. der Kommunikation.

Die wichtigsten Sorten von Anwenderprogrammen sind:

- Anlaufprogramm (bei Zustandswechsel der CPU von STOP zu RUN),
- Hauptprogramm (zyklisch bearbeitetes Anwenderprogramm) ,
- Alarmprogramm (Reaktion auf Alarm),
- Fehlerprogramm (Reaktion auf Fehler).

Die Schnittstellen zwischen Betriebssystem und Anwenderprogramm sind die Organisationsbausteine (OB) /Sys199/, /Sys299/. Sie sind Teil des Anwenderprogramms und werden ausschließlich vom Betriebssystem aufgrund eines aufgetretenen Startereignisses (Alarm, zyklisch, Fehler oder Hintergrund) gestartet und bearbeitet. Jeder OB-Start stellt dem Anwender eine systemeinheitliche Startinformation im Lokaldatenbereich zur Verfügung, aus der die Ursache des OB-Starts vom Programm ermittelt werden kann.

OB-Starts entstehen im wesentlichen aus folgenden Gründen:

- beim Anlauf der CPU,
- in zyklischer oder auch zeitlich getakteter Ausführung,
- zu bestimmten Uhrzeiten oder Tagen,
- nach Ablauf einer vorgegebenen Zeitdauer,
- beim Auftreten von Fehlern,
- beim Auftreten von Prozeß- oder Kommunikationsalarmen.

Die Organisationsbausteine sind in Prioritätsklassen eingeteilt, die beim Auftreten mehrerer Ereignisse die Reihenfolge der Programmbearbeitung festlegen. Jede OB-

Programmbearbeitung kann an Befehlsgrenzen von einem höherprioren Ereignis unterbrochen werden. Z. B. soll ein Alarmprogramm immer höherpriorer sein als das Hauptprogramm, damit die CPU in der Lage ist, sofort auf einen Alarm zu reagieren und den zugehörigen OB aufzurufen. In diesem OB steht die Routine, die eine entsprechende Reaktion auf den Alarm aktivieren soll. Das Programm wird abgebrochen, wenn die aktuelle Anweisung abgearbeitet ist, oder es verzweigt sich in eine Sonderbearbeitung und setzt den zyklischen Betrieb fort. Alle relevanten Daten werden gespeichert, damit nach der Bearbeitung des Alarmprogramms an der unterbrochenen Programmstelle die Bearbeitung des niederprioren Programms fortsetzt werden kann (Abbildung 2.2). OB gleicher Priorität unterbrechen sich nicht, sie werden in ihrer Erkennungsreihenfolge nacheinander bearbeitet.

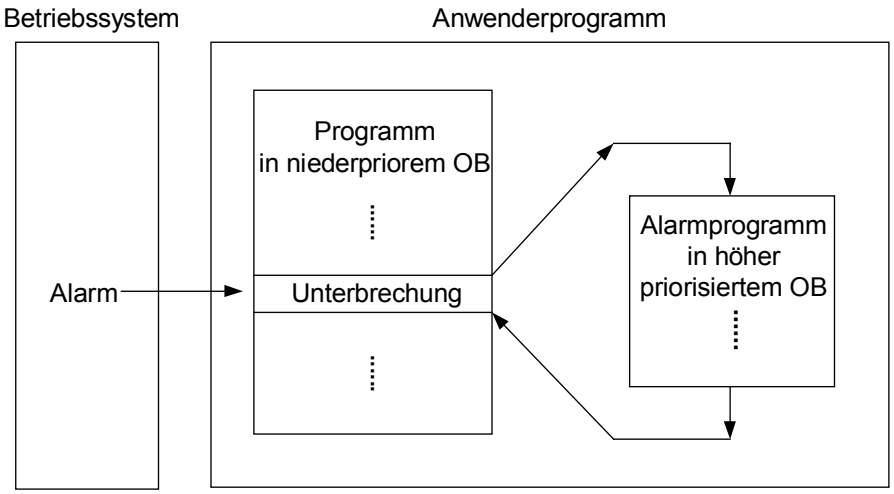


Abbildung 2.2: Aufrufsreihenfolge von OB

OB-Prioritäten und OB-Startereignisse können vom Anwender maskiert und geändert werden und zwar durch

- Konfiguration und Parametrierung,
- Programmiergerät- (PG-)Testfunktionen,
- das Anwenderprogramm.

2.2.2 Zyklische Programmbearbeitung

Ein wesentlicher Unterschied zwischen dem Programmablauf eines üblichen Rechners und eines Automatisierungssystems ist die Art der Abarbeitung eines Programms. In der konventionellen Datenverarbeitung wird ein Programm nur einmal ausgeführt, abgesehen von wenigen Ausfällen. Dagegen werden SPS-Programme fortlaufend bearbeitet. Das heißt, das Hauptprogramm wird erneut von oben nach unten zyklisch durchlaufen (Abbildung 2.3) /Frü97/.

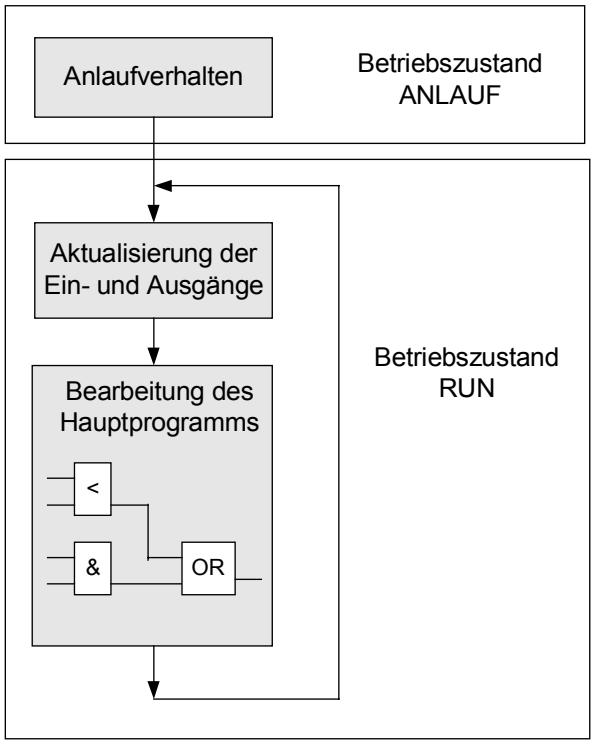


Abbildung 2.3: Zyklische Bearbeitung

Der zyklische Ablauf wird z.B. durch den niederprioren OB1 realisiert. Man könnte diesen Organisationsbaustein auch als Realisierung einer Idle-Loop ansehen, die die sämtliche Rechenleistung, die die höherprioren Dienste übriglassen, konsumiert. Man kann auch in diesem OB äquidistant periodische Abläufe implementieren, doch müssen sich dann alle Teilnehmer in der CPU, insbesondere die höherprioren, „kooperativ“ verhalten, d.h. die Kontrolle nur eine angemessen kurze Zeit an sich ziehen, damit auch niederpriore Tasks im OB1 regelmäßig abgearbeitet werden.

Vor Beginn jedes Zyklus wird der Zustand aller Eingänge in einen Puffer kopiert, der das Prozeßabbild der Eingänge konsistent für den Zyklus konserviert. Das Prozeßabbild ist ein interner Speicherbereich. Um Konsistenz zu gewährleisten, wird während der Abarbeitung des Programms nicht direkt auf die Eingänge zugegriffen, sondern immer nur auf die Daten in dem Puffer.

Analog werden auch während eines Zyklus die Ausgänge nicht sofort geschrieben, sondern in einen internen Puffer abgelegt, dem Prozeßabbild der Ausgänge. Am Ende eines Zyklus wird das Prozeßabbild komplett aus dem Puffer auf die physikalischen Ausgänge übertragen. Das Setzen eines Ausgangs wirkt sich also erst am Zyklusende aus.

Die zyklische Bearbeitung des OB1 wird nach Beendigung des Anlaufs der CPU begonnen. Die folgenden Ereignisse bewirken, daß das Betriebssystem den OB1 aufruft:

- Ende der Bearbeitung des Anlaufs,

- Ende der Bearbeitung des OB1 (des vorherigen Zyklus).

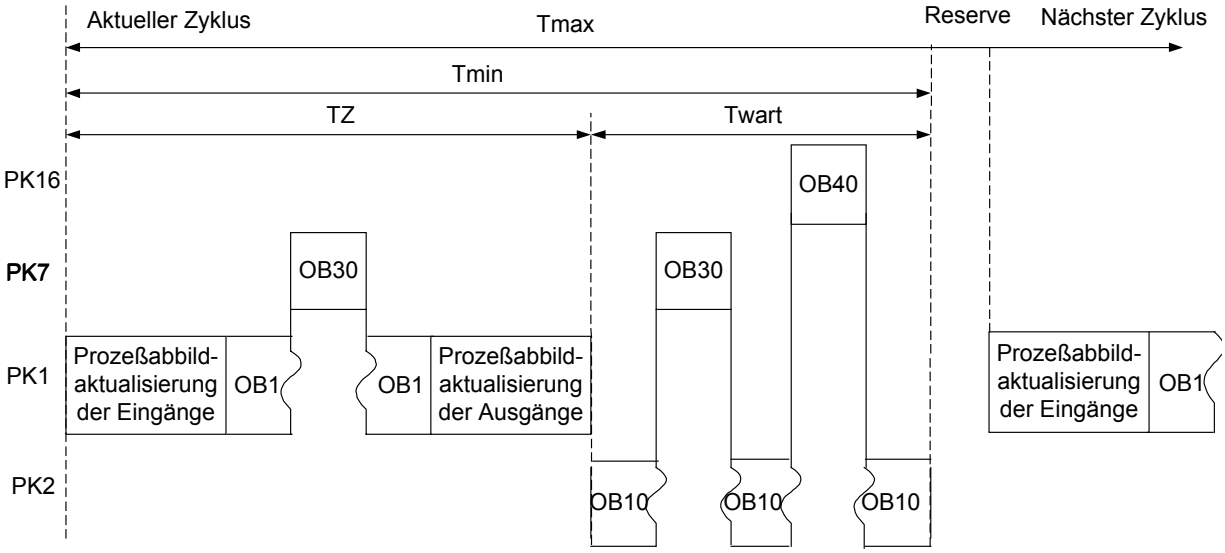
Falls eine maximale Reaktionszeit systemseitig in einer Grenze gehalten werden soll, kann man bei einigen Produkten (z.B. SIMATIC S7-400) eine Maximalzykluszeitüberwachung zwischen zwei aufeinanderfolgenden Zykluskontrollpunkten einführen, die defaultmäßig 150ms beträgt. Diese Überwachungszeit kann mit Hilfe eines System-Calls an beliebiger Stelle des Anwenderprogramms durch den Anwender "nachgetriggert", d.h. neu gestartet werden, um im speziellen Fall deren Ablauf zu verhindern.

Überschreitet die Abarbeitungszeit der Ablaufebene "Freier Zyklus" mit ihrer Systemprogrammbearbeitung und der Anwenderprogrammbearbeitung, die sich aus der Bearbeitungszeit des Freien Zyklus und der zwischenzeitlich eingeschachtelten höherprioren Ablaufebenen zusammensetzt, die Maximalzykluszeit, so wird als "asynchroner Fehler" ein sog. Zeitfehler des Freien Zyklus über den dafür vorgesehenen OB gemeldet.

Es ist umgekehrt auch möglich, eine Mindestzykluszeit einstellen. Dies ist sinnvoll, wenn

- die Zeitabstände zwischen den Starts der Programmbearbeitung des OB1 (Freier Zyklus) gleich lang sein sollen oder
- bei zu kurzer Zykluszeit die Aktualisierung der Prozeßabbilder unnötig oft erfolgen würde.

Abbildung 2.4 zeigt, wie die Zykluszeiten eingehalten und OBs je nach Prioritätsklasse ausgeführt werden. Je größer die Prioritätsklasse ist, desto bevorzugter wird dieser OB aufgerufen.



Tmax = die einstellbare Maximalzykluszeit
Tmin = die einstellbare Mindestzykluszeit
TZ = die tatsächliche Zykluszeit
Twart = die Differenz zwischen Tmin und tatsächlicher Zykluszeit, in dieser Zeit können auftretende Alarme sowie der Hintergrund-OB bearbeitet werden
PK = Prioritätsklasse

Abbildung 2.4: Zyklusüberwachung

2.3 Automatisierung mit SIMATIC

Das Automatisierungspaket SIMATIC / PCS7 der Firma Siemens ist ein Prozeßleitsystem, das aus Automatisierungsrechnern, Programmiergeräten, IndustriePC, Speicherprogrammierbaren Steuerungen, Komplettgeräten, Bedien- und Beobachtungssystemen, Kommunikationsnetzen, dezentraler Peripherie und Industriesoftware bestehen kann. Die Komponenten von Prozeßleitsystemen sind aufeinander abgestimmt mit einheitlicher Projektierung, Datenhaltung und Datenübertragung. SIMATIC wird sowohl in der Fertigungsindustrie als auch in der Prozeßindustrie weltweit eingesetzt. Es wird im Rahmen des experimentellen Anteils dieser Arbeit benutzt.

2.3.1 Bedienen und Beobachten

Von einer Operator Station (OS) aus kann das Bedienungs-, Wartungs- und Betriebspersonal den Prozeß verfolgen bzw. beeinflussen, d.h. Rezepte oder Batch-Abläufe modifizieren, aktuelle Werte ändern oder mit dem Prozeß über die Automatisierungssysteme kommunizieren. Außerdem werden auch Alarme und Bedienanforderungen hier bearbeitet.

SIMATIC HMI (Human Machine Interface) stellt die benötigten Werkzeuge zur Verfügung. Die Bedien- und Beobachtungsgeräte werden mit dem Softwarepaket WinCC projektiert und können über MPI oder PROFIBUS mit dem SPS Daten austauschen. SIMATIC WinCC ist ein branchen- und technologieneutrales System zur Lösung von Visualisierungs- und leittechnischen Aufgaben in der Produktions- und Prozeßautomatisierung.

2.3.2 Engineering

SIMATIC bietet ein system- und anlagenweites Engineering für alle Komponenten des gesamten Systems: Operator Stationen, Automatisierungssysteme und Dezentrale Peripherie /PCS97/. Aufsetzend auf der systemweiten und konsistenten Datenhaltung steht zur Projektierung ein komplettes Engineering Toolset zur Verfügung, bestehend aus STEP 7, SCL, CFC, DOCPRO und leittechnischen Optionen wie SFC, Technologische Hierarchie und Import-/Export-Assistent.

STEP 7 Basispaket

Die in STEP 7 integrierten SIMATIC-Programmiersprachen erfüllen die Norm DIN EN 6.1131-3. Das Basispaket läuft unter dem Betriebssystem Windows 95/98/NT und ist an dessen grafische und objektorientierte Arbeitsweise angepaßt.

Das Basispaket unterstützt den Anwender in allen Phasen des Erstellungsprozesses von Automatisierungslösungen, wie z.B.:

- Einrichten und Verwalten von Projekten,

- Konfigurieren und Parametrieren der Hardware und der Kommunikation,
- Verwaltung von Symbolen,
- Programmerstellung z. B. für S7-Zielsysteme,
- Laden von Programmen auf Zielsysteme,
- Test der Automatisierungsanlage,
- Diagnose bei Anlagenstörungen.

Für diese Unterstützungen stellt das STEP 7-Basispaket dem Anwender eine Reihe von Tools zur Verfügung (Abbildung 2.5).

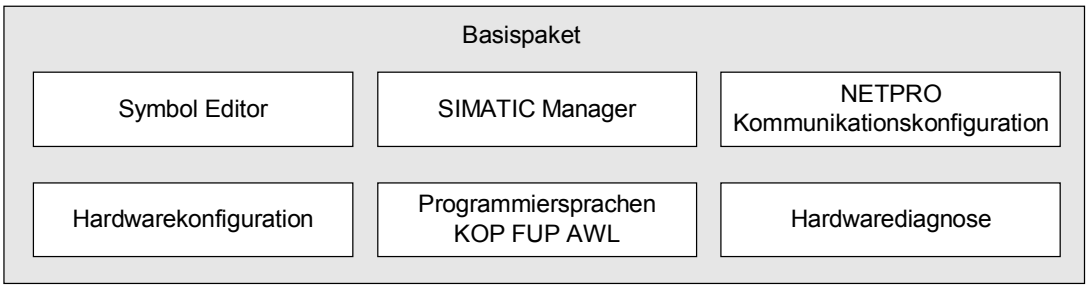


Abbildung 2.5: STEP7 Basispaket

CFC (Continuous Function Chart)

Der CFC (Continuous Function Chart) ist ein grafischer Editor, der auf dem Software-Paket STEP 7 aufsetzt /CFC98/. Er dient dazu, aus vorgefertigten Bausteinen eine Gesamt-Software-Struktur für eine CPU zu erstellen. Hierzu werden Bausteine auf Funktionspläne plaziert, parametriert und verschaltet.

Verschalten bedeutet, daß für die Kommunikation zwischen Bausteinen oder globalen Operanden Verbindungen hergestellt werden, damit Werte von einem Ausgang zu einem oder mehreren Eingängen übertragen werden können.

Der CFC kann für die Projektierung verschiedener Zielsysteme verwendet werden, z.B. die Automatisierungssysteme SIMATIC S7, SIMATIC M7, SIMADYN D, ... und inzwischen auch Fremdprodukte z.B. der Firmen Landis/Steffa und Klöckner/Möller.

SFC (Sequential Function Chart)

Ein SFC-Plan ist eine Ablaufsteuerung. Der SFC-Editor ist ein Werkzeug zum Erstellen einer Ablaufsteuerung /SFC97/. Eine Ablaufsteuerung ist eine Steuerung mit zwangsläufig schrittweisem Ablauf, die von einem Zustand abhängig von Bedingungen zum nächsten Zustand weiter schaltet.

Mit Ablaufsteuerungen können z.B. die Herstellungsvorschriften von Produkten als ereignisgesteuerte Prozesse beschrieben werden (Rezepte). Mit einer Ablaufsteuerung

werden (typischerweise mit CFC erstellte) Funktionen der Basisautomatisierung per Betriebs- und Zustandswechsel gesteuert und selektiv bearbeitet.

Die typischen Einsatzbereiche für Ablaufsteuerungen liegen im Bereich der Anlagen mit diskontinuierlicher Betriebsweise. Aber auch für kontinuierlich arbeitende Anlagen können Ablaufsteuerungen eingesetzt werden, z.B. für An- und Abfahrvorgänge, Arbeitspunktänderungen sowie Zustandswechsel bei Störungen usw..

Programmiersprachen KOP, FUP, AWL und SCL

Die Programmiersprachen sind nach der Norm DIN EN 6.1131-3 (IEC 1131-3) ausgelegt. Das Anwenderprogramm kann mit diesen Programmiersprachen erstellt werden.

KOP und FUP sind grafisch orientierte Sprachen, in KOP wird die Steuerungsaufgabe durch die Reihen- und Parallelschaltung von Kontakten gelöst und in FUP durch die Verschaltung von UND- und ODER-Boxen. AWL ist textorientiert; hier geschieht die Beschreibung der Steuerungsaufgabe in Listenform /Anw98/. SCL ist eine PASCAL-artige Hochsprache, optimiert für die Programmierung von speicherprogrammierbaren Steuerungen.

Die grafischen Programmiersprachen eignen sich besonders zur Darstellung binärer Verknüpfungen, sei es als Nachbildung eines Stromlaufplans (Kontaktplan KOP) oder als Nachbildung elektronischer Schaltkreise (Funktionsplan FUP). Die textuelle Sprache Anweisungsliste AWL eignet sich zusätzlich zum Bearbeiten komplexer Variablen oder zur indirekten Operandenadressierung. SCL ist auch eine textuelle Programmiersprache zur Umsetzung komplexer Algorithmen und zur Handhabung großer strukturierter Datenmengen.

Import-/Export-Assistent

Das Engineeringssystem (ES) ist in die bestehende Infrastruktur der Gesamtanlagen-Konfigurierung einbindbar. Bereits in frühen Planungsphasen und in anderen CAE-Systemen entstandene Daten, z.B. in Form von Meßstellenlisten aus der Funktions- und Anlagenplanung, können in das ES übernommen und weiterverarbeitet werden. Mit dem Import-/Export-Assistenten können für Funktionseinheiten, z.B. eine Kaskadenregelung, Musterlösungen vorgefertigt werden, die später beliebig oft kopiert und für die konkrete Lösung modifiziert werden können.

OS-Engineering

Die Projektierung der Operator Stationen (OS) wird ebenfalls über den SIMATIC Manager organisiert. Dazu können in der Systemkomponentensicht OS eingerichtet werden. In der technologischen Sicht können OS-Objekte, z.B. Fließbilder, funktional zugeordnet werden.

Alle OS-spezifischen Projektierungen werden mit den dafür vorgesehenen Mitteln durchgeführt. So werden z.B. die Fließbilder mit dem Grafik-Editor erstellt und Aktionen, Archive, Protokolle und Kurven projektiert.

BATCH Betrieb

Das Softwarepaket BATCH flexible bietet für die Verfahrensindustrie einen Einstieg in den rezeptgesteuerten Batch-Betrieb. Neben SFC aus dem ES zur Erstellung von Ablaufsteuerungen wird der Anwender zusätzlich bei rezeptgesteuerten Chargenprozessen unterstützt.

Dokumentation

Mit dem Dokumentationswerkzeug DOCPRO kann folgendes erstellt werden:

- Dokumentation der Anlagendaten,
- benutzerspezifische Anlagendokumentation für die Automatisierung.

Bibliotheken

Bibliotheken sind Gruppen vorgefertigter Bausteine, die nach Funktionalität (anwenderspezifische Gesichtspunkte) gegliedert und zusammengefaßt sind. Bei der Projektierung ist dann nur eine Auswahl von Bausteinen erforderlich. Die Projektierungsarbeit wird dadurch erheblich erleichtert.

2.3.3 Automatisierungssystem und dezentrale Peripherie

Die SPS SIMATIC S7 bilden die Basis der SIMATIC-Hardware. Es gibt drei Bauformen, SIMATIC S7-200, SIMATIC S7-300 und SIMATIC S7-400, die jeweils für den unteren, mittleren und gehobenen Leistungsbereich geeignet sind.

Automatisierungssysteme sind modular aufgebaut. Sie bestehen aus Baugruppenträger, CPU, Stromversorgung sowie einer Anschaltung an den Systembus. Als Beispiel sind zwei komplette Automatisierungssysteme zu erwähnen, AS414 mit 384 kbyte Arbeitsspeicher und AS416 mit 800 oder 1600 kbyte Arbeitsspeicher. AS414 ist speziell für kleinere Anwendungen mit geringeren Mengengerüsten zugeschnitten. Sie erfüllen damit die Forderungen nach einem skalierbaren und modularen System, das für kleinere Anwendungen einen kostengünstigen Einstieg bietet. AS416 wird dagegen dann eingesetzt, wenn größere Mengengerüste in größeren Anlagen benötigt werden.

Für die Peripherieanbindung sind beispielsweise die dezentralen ET-200M-Komponenten im Einsatz. Diese werden über den Feldbus PROFIBUS-DP an das Automatisierungssystem angebunden. Daraus ergibt sich eine hohe Flexibilität beim Aufbau der I/O-Peripherie sowohl zentral in Elektronikräumen als auch dezentral in vorgelagerten Schalträumen. Aus Gründen erhöhter Verfügbarkeit ist es möglich, im laufenden Betrieb Baugruppen auszuwechseln und umzubauen, ohne das betreffende ET-200M-Subsystem abschalten zu müssen.

Neben den Baugruppen der dezentralen Peripherie ET-200M lassen sich über den PROFIBUS-DP auch Feldgeräte anschließen. Dies sind u.a. Stellantriebe SIPOS, drehzahlgeregelte Antriebe SIMOVERT, Motor-Schutz- und Steuerbausteine SIMOCODE

und Kompaktregler SIPART DR19 und DR21 sowie auch alle PROFIBUS-fähigen Feldgeräte anderer Hersteller.

2.4 Aufbau eines Automatisierungssystems SIMATIC S7-400

In dieser Arbeit wird für alle Untersuchungen ausschließlich das Automatisierungssystem SIMATIC S7-400 der Siemens AG eingesetzt. Im folgenden wird die Zusammensetzung von diesem Automatisierungssystem genauer beschrieben.

Automatisierungssysteme werden heute von verschiedenen Herstellern angeboten. Alle für die Steuerungstechnik benötigte Verknüpfungsglieder, Speicherfunktionen, Zeiten, Zähler usw. sind vom Hersteller integriert und werden über die Programmierung zu einer funktionierenden Steuerung verbunden. Es gibt eine Vielzahl von Steuergeräten, die sich durch folgende Funktionseinheiten unterscheiden:

- Ein- und Ausgänge,
- Speicherplätze,
- Zähler,
- Zeiten,
- Merkerfunktionen,
- Sonderfunktionen,
- Arbeitsgeschwindigkeit,
- Art der Programmbearbeitung.

Größere Steuergeräte werden aus einzelnen Baugruppen in Modulbauweise individuell zusammengesetzt. Mit diesem modularen System lassen sich, von der Grundausstattung ausgehend, SPS-Systeme zusammenstellen, die für den Anwendungsfall angepaßt werden können. Für kleinere Steuerungsaufgaben werden kompakt aufgebaute Steuergeräte angeboten. Sie sind in sich geschlossene Einheiten, mit einer fest vorgegebenen Anzahl von Ein- und Ausgängen.

2.4.1 Komponenten des Automatisierungssystems SIMATIC S7-400

Ein komplettes Automatisierungsgerät einschließlich aller Peripheriebaugruppen nennt man eine "Station". Eine S7-400-Station besteht mindestens aus einem Baugruppenträger mit Stromversorgung und CPU. Die Kopplung zur Maschine oder Anlage übernehmen Peripheriebaugruppen (Abbildung 2.6) /Aut99/.

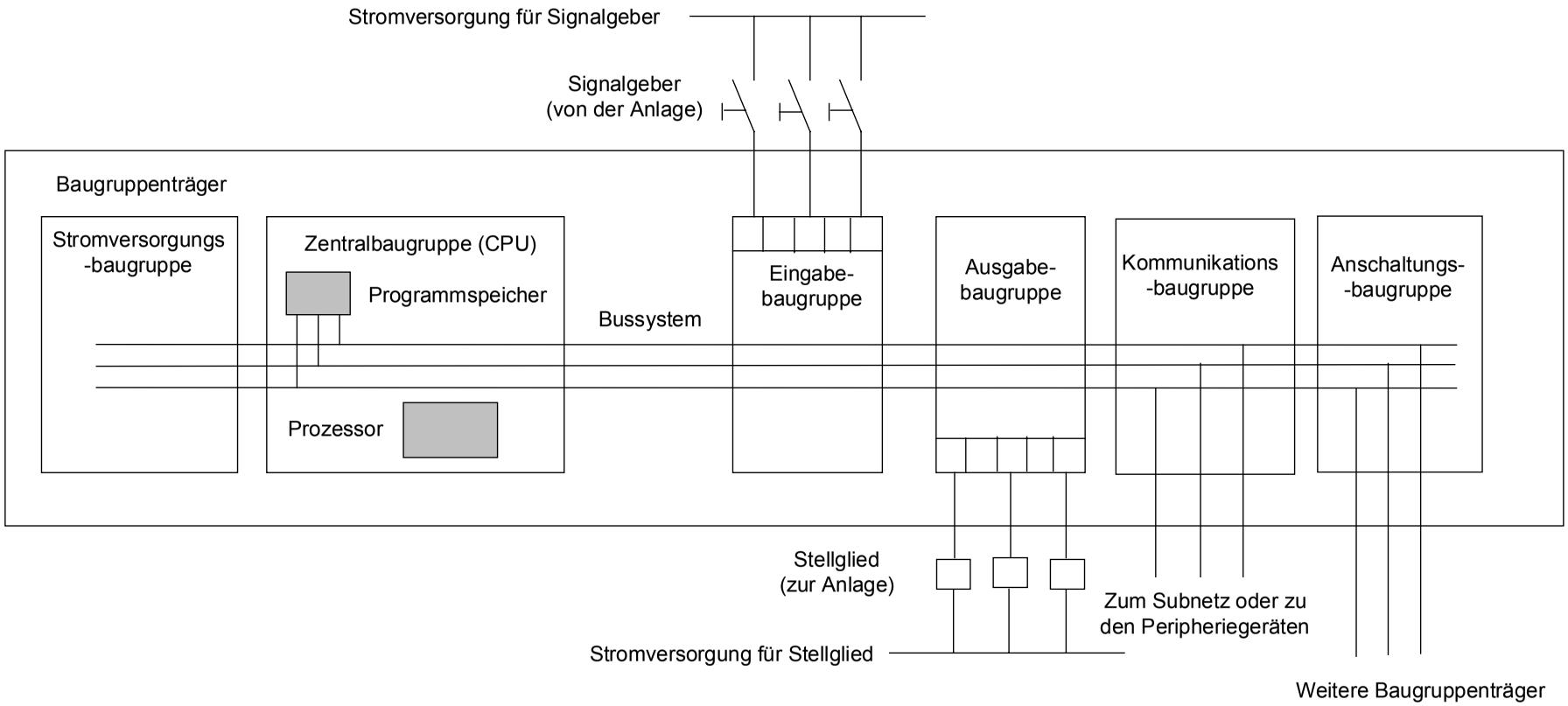


Abbildung 2.6: Komponenten eines Automatisierungssystems

Die folgende Auflistung zeigt, welche Komponenten eine SIMATIC-Station in den meisten Anwendungen enthält:

- Baugruppenträger (Racks),
- Stromversorgungsbaugruppe (PS, power supply),
- Anschaltungsbaugruppe (IM, interface module),
- Zentralbaugruppe (CPU, central processor unit),
- Signalbaugruppen (SM, signal module), Eingang und Ausgang,
- Kommunikationsbaugruppen (CP, communication processor),
- Peripheriegeräte.

2.4.2 Baugruppenträger, Stromversorgungsbaugruppe und Anschaltungsbaugruppe

Baugruppenträger nehmen die Baugruppen auf und verbinden sie untereinander. Die Baugruppenträger bei S7-400 können als Zentralbaugruppenträger (CR, central rack), Erweiterungsbaugruppenträger (ER, extension rack) oder für beides (UR, universal rack) eingesetzt werden. Sie bestehen aus einer Aluminiumprofilschiene, haben jeweils eine feste Länge und sind mit dem Rückwandbus und den Bussteckern komplett ausgestattet.

Der Rückwandbus in einem Baugruppenträger ist zweigeteilt (Abbildung 2.7): Der P-Bus (Peripheriebus) ist für den schnellen Austausch von Ein-/Ausgabesignalen optimiert, der K-Bus (Kommunikationsbus) für den Austausch größerer Datenmengen. Über die Verbindung von K-Bus und MPI-Schnittstelle der CPU sind die FM- und CP-Baugruppen mit K-Busanschluß an das MPI-Bussystem angeschlossen; sie können so beispielsweise über die Programmiergeräteschnittstelle der CPU parametriert werden.

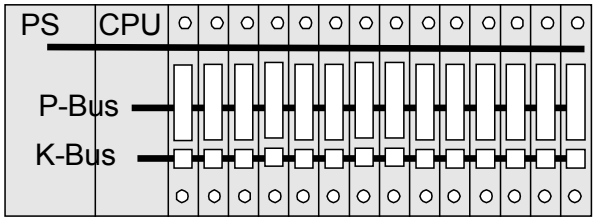


Abbildung 2.7: Universalbaugruppenträger

Anschaltungsbaugruppen verbinden die Baugruppenträger untereinander. Der für den Betrieb notwendige Strom wird durch die Stromversorgungsbaugruppe bereitgestellt. Sie wandelt die Netzspannung von 120/230 V AC in eine Gleichspannung 24 V DC für die Versorgung der S7 Baugruppen um und befindet sich normalerweise an den ersten beiden Steckplätzen auf dem Baugruppenträger.

2.4.3 Zentralbaugruppe

Die SIMATIC S7-CPU-Baugruppen enthalten den zentralen Steuerungsprozessor, der das Betriebssystem enthält, das Anwenderprogramm abarbeitet, die Baugruppen parametriert und die Nutzdaten überträgt. Alle CPU-Baugruppen besitzen eine MPI-Schnittstelle für den Anschluß an das Programmiergerät oder für den Anschluß an Bedien- und Beobachtungsgeräte oder andere SIMATIC-Stationen.

Betriebszustand

Das Betriebssystem regelt alle Vorgänge auf der CPU-Baugruppe. Es steuert den Einschaltvorgang, stößt die Bearbeitung des Hauptprogramms an, aktualisiert die Prozeßabbilder, erfaßt Alarme und Fehlermeldungen und veranlaßt die Bearbeitung der dazugehörenden Ablaufebenen, überwacht schließlich alle Vorgänge und meldet erfaßte Fehler mit Eintrag in den Diagnosepuffer und mit LED an der Frontseite der Baugruppe. Hier befindet sich auch der Betriebsartenschalter, mit dem die Bearbeitung des Anwenderprogramms eingeschaltet wird.

Der Betriebsartenschalter kennt 4 Stellungen: STOP, RUN, RUN-P und die Taststellung MRES. Mit STOP wird die Bearbeitung des Anwenderprogramms angehalten; RUN und RUN-P schalten die Bearbeitung ein. In der Stellung RUN kann das Anwenderprogramm nur gelesen und in RUN-P auch verändert werden. RUN-P garantiert ein Laden der geänderten Bausteine mit größtmöglicher Sicherheit (stoßfrei). Die Stellung MRES wird für Urlöschen und den Kaltstart benötigt.

SIMATIC S7-400 kann sich auch nur in vier unterschiedlichen Betriebszuständen befinden: ANLAUF, HALT, RUN, STOP.

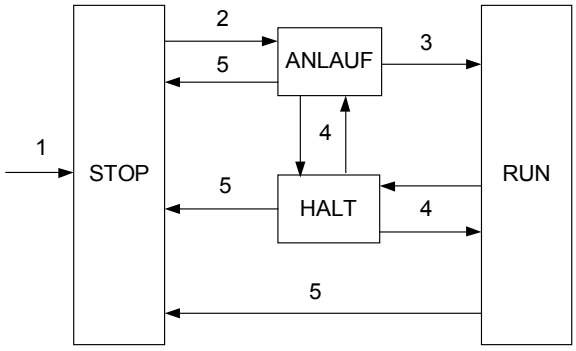


Abbildung 2.8: Betriebsarten einer SIMATIC S7-CPU

In der Abbildung 2.8 sind die möglichen Wechsel von einem Zustand in einen anderen der CPU eingezeichnet. Nach dem Einschalten (1) befindet sich die CPU im Zustand STOP. Dreht man den Betriebsartenschalter auf RUN oder RUN-P, wird zuerst ein Anlaufprogramm durchlaufen (2), bevor die eigentliche Bearbeitung des Anwenderprogramms mit RUN beginnt (3). Im ANLAUF und im RUN können Testfunktionen durchgeführt werden, die in den

Betriebszustand HALT führen (4). Tritt in einem dieser Betriebszustände ein Fehler auf, fällt die CPU in den Zustand STOP zurück (5).

Logische Trennung eines STEP7-Programms in Prozeß- und Kommunikationsteil

Eine S7-CPU hat folgende Grundaufgaben zu erfüllen:

- Automatisierungsaufgaben im Sinne von Messen, Steuern, Regeln, Diagnostizieren und Leiten,
- Schnittstelle zu datentechnisch orientierten Aufgaben,
- Kommunikationsaufgaben zur Unterstützung der obengenannten Aufgaben.

Der Aufgabenstellung folgend besteht eine S7-CPU aus zwei logischen Teilen (Abbildung 2.9),

- dem Kommunikationsteil und
- dem Prozeßteil.

Der Prozeßteil ist in erster Linie für das Abarbeiten eines STEP 7 Programms zuständig. Er bedient am Rückwandbus der CPU den P-Bus.

Der Kommunikationsteil bearbeitet kein Anwenderprogramm. Er nimmt Kommunikationsaufträge von verschiedenen Teilen an und leitet sie weiter. Der Kommunikationsteil ist für das Bedienen der MPI, des K-Busses, der weiteren integrierten Schnittstellen (z.B. der integrierten DP-Schnittstellen) und des Ladespeichers zuständig. Er arbeitet asynchron zum Prozeßteil.

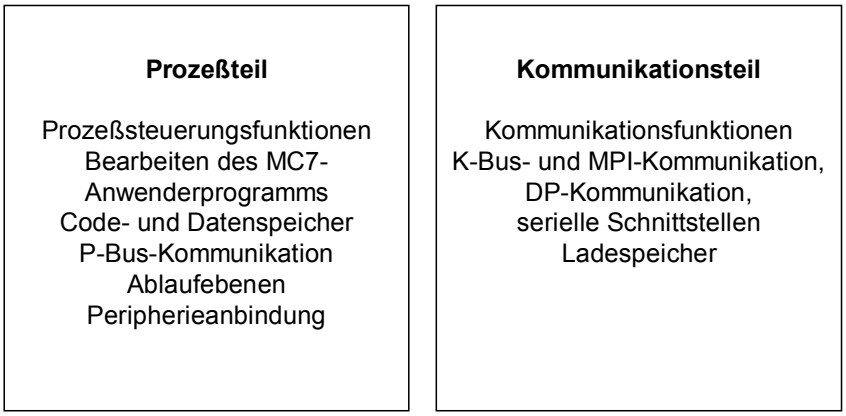


Abbildung 2.9: Aufgaben im Prozeßteil und im Steuerungsteil

S7-400-CPUs bearbeiten das Anwenderprogramm direkt mit einem speziell dafür ausgelegten ASIC. Die Bearbeitungszeit für 1000 Binäranweisungen liegt hier zwischen ca. 0,2 ms für die CPU412 und ca. 0,08 ms für die CPU416. Die Kommunikation mit dem Programmiergerät oder den Datenaustausch mit anderen Stationen übernimmt bei der S7-400 ein eigener Kommunikationsprozessor. Dies ermöglicht das asynchrone parallele

Abarbeiten der Kommunikationsaufgaben. Die asynchrone Belastung des Prozeßteils durch Kommunikationsaufgaben entfällt dadurch. Die maximale Belastung des Prozeßteils durch Kommunikationsaufgaben (synchrone Belastung + asynchrone Belastung) kann in der CPU-Parametrierung eingestellt werden.

Anwenderspeicher

Es gibt auf der CPU zwei Arten von Speicher, Ladespeicher und Arbeitspeicher, für das Anwenderprogramm. Er kann auf der Baugruppe integriert sein oder als zusteckbare Memory Card ausgeführt.

Das Programmiergerät überträgt das komplette Anwenderprogramm einschließlich der Konfigurationsdaten in den Ladespeicher. Das Betriebssystem interpretiert die Konfigurationsdaten, parametriert die CPU-Baugruppe und – im Anlauf – die Peripheriebaugruppen. Der ablaufrelevante Programmcode und die Anwenderdaten werden in den Arbeitsspeicher kopiert, welcher normalerweise ein schneller RAM-Speicher ist. Beim Zurückladen des Anwenderprogramms in das Programmiergerät werden die Bausteine aus dem Ladespeicher geholt, ergänzt um die aktuellen Werte der Datenoperanden aus dem Arbeitsspeicher.

Falls man den Ladespeicher erweitern möchte, kann Memory Card eingesetzt werden. Es gibt zwei Arten von Memory Cards: RAM Memory Cards und Flash EPROM Memory Cards. Wenn ausschließlich der Ladespeicher erweitert werden soll, verwendet man ein RAM Memory Card. Aber falls das Anwenderprogramm einschließlich der Konfigurationsdaten und Baugruppenparameter auch ohne Pufferbatterie spannungsausfallsicher gehalten werden soll, verwendet man ein Flash EPROM Memory Card.

Operandenbereiche

Nicht zum Anwenderspeicher gehört der Systemspeicher, in dem Systemdaten oder Operanden (Variable) gespeichert werden. Die Operanden sind zu Bereichen (Operandenbereiche) zusammengefaßt, die eine für jede CPU spezifische Menge an Operanden enthalten. Folgende Operandenbereiche liegen im Systemspeicher der CPU (Abbildung 2.10):

- Eingänge (E): Sie sind ein Abbild ("Prozeßabbild") der Digitaleingabebaugruppen.
- Ausgänge (A): Sie sind ein Abbild ("Prozeßabbild") der Digitalausgabebaugruppen.
- Merker (M): Sie sind Informationsspeicher, die im gesamten Programm ansprechbar sind.
- Zeitfunktionen (T): Sie stellen Zeitglieder dar, mit denen Warte- und Überwachungszeiten realisiert werden.
- Zählfunktionen (Z): Sie sind Softwarezähler, die vorwärts und rückwärts zählen können.
- temporäre Lokaldaten (L): Sie dienen als dynamische Zwischenspeicher während der Bausteinbearbeitung und stehen im L-Stack, den die CPU während der Programmbearbeitung dynamisch belegt.

Zusätzlich enthält der Systemspeicher Puffer für Kommunikationsaufträge und Systemmeldungen (Diagnosepuffer). Die Größe dieser Datenpuffer sowie die Größe des Prozeßabbilds und des L-Stacks sind bei der S7-400-CPUs parametrierbar.

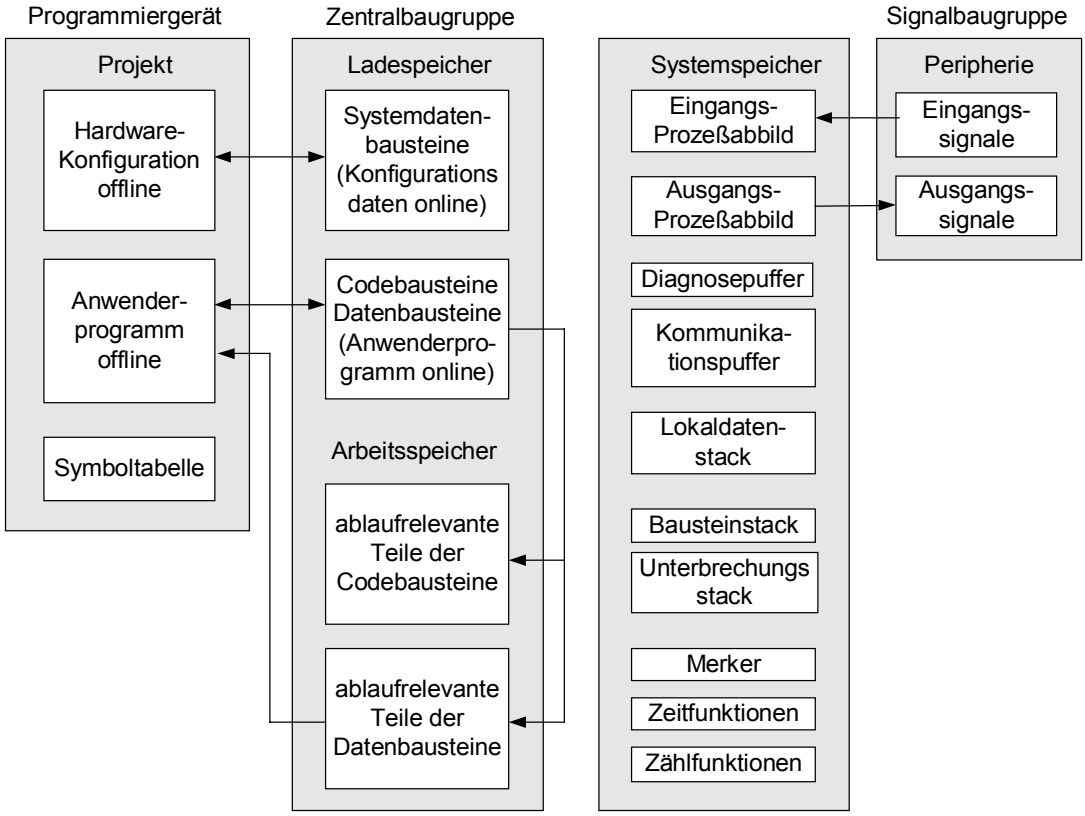


Abbildung 2.10: Speicherbereiche für Anwenderprogramm und Operanden

2.4.4 Signalbaugruppen

Signalbaugruppen passen die Signale der gesteuerten Anlage an den internen Signalpegel an oder steuern Schütze, Stellgeräte, Leuchten, usw. Signalbaugruppen gibt es als Ein- und Ausgabebaugruppen für Digital- und Analogsignale, auch für den Anschluß von Sensoren und Aktoren (z.B. aus explosionsgefährdeten Anlagen).

Digitalbaugruppen sind Signalformer für binäre Prozeßsignale. Mit Eingabebaugruppen kann die CPU die Betriebszustände des Prozesses abfragen und mit Ausgabebaugruppen steuernd in den Prozeß eingreifen. Die Potentialtrennung der Digitalsignale zwischen Rückwandbus und Prozeß erfolgt mittels Optokoppler. Es gibt Digitalbaugruppen mit ein, zwei oder vier Bytes entsprechend 8, 16 oder 32 Signalen. Die Adressierung von Digitalbaugruppen erfolgt bevorzugt im Prozeßabbild, so daß die Signalzustände bitweise verarbeitet werden können. Komfortabel ausgestattete Digitalbaugruppen können Diagnosedaten über den Baugruppenzustand abgeben.

Analogbaugruppen sind Signalformer für analoge Prozeßsignale. Mit Hilfe dieser Baugruppen kann die CPU analoge Meßgrößen verarbeiten, nachdem sie von Analogeingabebaugruppen in digitale Werte umgewandelt worden sind, oder Stellgeräte kontinuierlich mit analogen Sollwerten versorgen, die von Analogausgabebaugruppen aus den digital vorgegebenen Werten erzeugt werden. Je analoge Größe (z.B. Meßwert oder Sollwert) wird auf der Baugruppe ein "Kanal" belegt. Es gibt Analogbaugruppen mit 4, 8 oder 16 Kanälen entsprechend 8, 16 oder 32 Bytes. Intern wird ein digitalisierter Analogwert als 16-Bit-Ganzzahl dargestellt. Komfortabel ausgestattete Analogbaugruppen können Diagnosedaten über den Baugruppenzustand oder das Erreichen eines Grenzwerts abgeben.

2.4.5 Kommunikationsbaugruppen

Kommunikationsbaugruppen entlasten die CPU von Kommunikationsaufgaben. Sie verbinden die SIMATIC-Station mit Subnetzen, wie z.B. Industrial Ethernet, PROFIBUS-FMS, AS-Interface oder serieller Punkt-zu-Punkt-Kopplung, übernehmen den Verbindungsaufbau und den Datentransport über das Netz und stellen die dafür erforderlichen Kommunikationsdienste der CPU und dem Anwenderprogramm zur Verfügung.

2.4.6 Peripheriegeräte

Zu einer Station zählt auch die an diese Station angeschlossene dezentrale Peripherie. Zur Anbindung der Peripheriegeräte wird der standardisierte Feldbus PROFIBUS-DP eingesetzt.

Eine Station kann mehrere DP-Master enthalten, die über ein oder mehrere PROFIBUS-DP-Subnetze die DP-Slaves und damit die Feldgeräte steuern. Die DP-Slaves sind in den Adreßraum der zentralen Peripherie eingebunden und werden auch weitgehend wie die Peripheriebaugruppen in den Zentral- und Erweiterungsbaugruppenträgern angesprochen.

Anbindbar sind die dezentrale ET 200M und alle normkonformen PROFIBUS-Feldgeräte. Mit ET 200M lassen sich die Signalbaugruppen ebenfalls anschließen. Somit kann der Aufbau der Signalbaugruppen sowohl zentral als auch dezentral stattfinden.

Mit den beiden Netzkomponenten DP/PA Koppler und DP/PA Link sind auch Feldgeräte mit der PROFIBUS-PA-Schnittstelle anschließbar. Zusätzlich kann auch ein AS-Interface verbunden werden, das ein Vernetzungssystem für den direkten Anschluß einfacher binärer Sensoren und Aktoren ist.

3 Kommunikationstechnisches Umfeld

Kommunikationsnetze im Bereich industrieller Anwendung haben viele Gemeinsamkeiten mit denen im Bereich der Büroanwendungen, z.B. Netztopologie. Aufgrund der industriellen Anforderung weist ein typisches Netzwerk in der Automatisierungstechnik jedoch auch eigene Besonderheiten auf. Bezüglich einer schnellen Reaktionszeit ist es z.B. meistens notwendig, ein vereinfachtes Modell einzusetzen, um die Bearbeitungszeit zu reduzieren. Außerdem werden viele Kommunikationsdienste für unterschiedliche Zwecke angeboten. Das kommunikationstechnische Umfeld wird im weiteren genauer beschrieben.

3.1 Netztopologie bei leittechnischen Anwendungen

Wenn mehrere eigenständige Automatisierungskomponenten, z.B. Sensoren, Aktoren, SPS, Informationen austauschen, dann müssen sie zwangsläufig in irgendeiner Struktur physikalisch miteinander verbunden sein. Sie bilden dann zusammen ein Kommunikationsnetzwerk. Unter der Netzwerktopologie versteht man die prinzipielle geometrische Struktur des Netzwerkes. Die Kommunikationsteilnehmer sind die Knoten des Netzwerkes, sie sind durch Kanten verbunden. Die am häufigsten vorkommenden Topologien haben eine Linien-, Ring-, Stern- oder Baumstruktur (Abbildung 3.1).

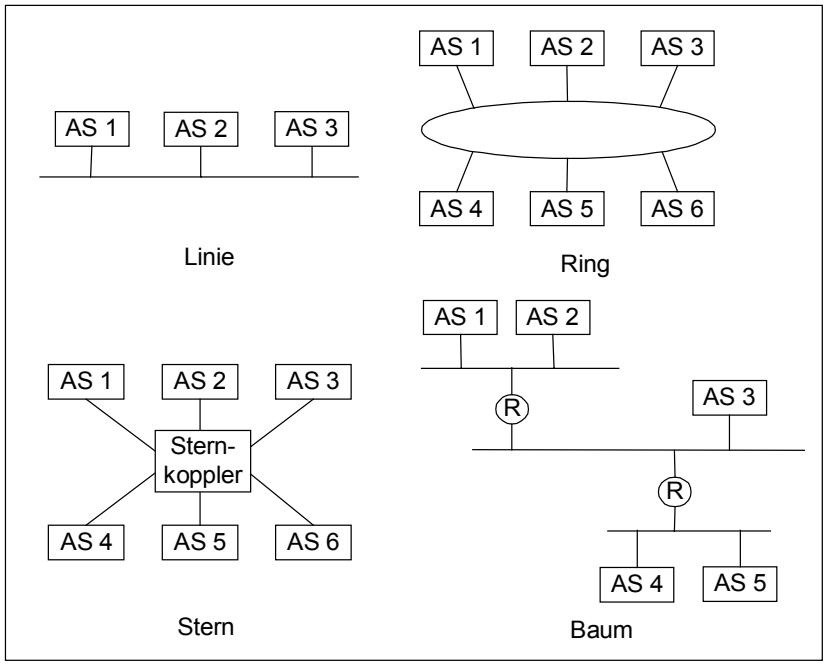


Abbildung 3.1: Netztopologie

3.2 Schichtenmodell

3.2.1 ISO/OSI Referenzmodell

Überall, wo Systeme kommunizieren, bedarf es der Abstimmung mit dem Kommunikationsmedium. Die betrifft nicht nur die physikalische Anbindung, sondern auch die Sprache, mit der sich die Systeme verständigen, d.h. das Protokoll der Kommunikation /Tan97/.

Aus diesem Grund wurde von der International Standardization Organisation ein 7-Schichtenmodell definiert /ISO/ (Abbildung 3.2). Jede einzelne Schicht legt jeweils ein zwischen den angeschlossenen Geräten vereinbartes, schichtspezifisches Protokoll fest. Die 7 Schichten sind hierarchisch angeordnet, die während der Kommunikation von den Nachrichten nacheinander von oben nach unten bzw. umgekehrt logisch zu durchlaufen sind. Dabei stellt jede Schicht ihrer oberen Nachbarschicht Leistungen, auch "Services" oder "Dienste" genannt, zur Verfügung und nimmt ihrerseits die Dienste der unteren Nachbarschicht in Anspruch, um ihre Aufgaben zu erledigen.

Die einzelnen Schichten sind in der Abbildung 3.2 festgelegt aufgelistet.

Schicht	Bezeichnung
7	Anwendungsschicht
6	Darstellungsschicht
5	Sitzungsschicht
4	Transportschicht
3	Vermittlungsschicht
2	Sicherungsschicht
1	Physikalische Schicht

Abbildung 3.2: ISO/OSI-Referenzmodell

Die Aufgaben der einzelnen Schichten werden im folgenden in aufsteigender Reihenfolge kurz beschrieben:

Schicht 1: Bitübertragungsschicht

Diese Schicht sorgt für die transparente Übertragung von Bits über das physikalische Medium in der Reihenfolge, wie sie von der Sicherungsschicht (Schicht 2) übergeben werden. Hier sind die elektrischen und mechanischen Eigenschaften sowie die Übertragungsarten festgelegt.

Schicht 2: Sicherungsschicht

Diese Schicht hat die Aufgabe, die Übertragung von Bitstrings über eine bereits existierende physikalische Verbindung zwischen zwei Systemen sicherzustellen.

Zu den Aufgaben gehört folgendes:

- Bildung der über das physikalische Medium zu übertragenden Telegramme,
- Erkennung und Behebung beziehungsweise Weitermeldung von Übertragungsfehlern,
- Flusskontrolle,
- Zugriffskontrolle.

Schicht 3: Vermittlungsschicht

Die Vermittlungsschicht dient vor allem zur Vermittlung von Daten zwischen den Endsystemen. Als Endsysteme sind der Sender und Empfänger einer Nachricht anzusehen, deren Weg unter Umständen über mehrere Transitsysteme führt. Dazu ist von der Vermittlungsschicht eine Wegwahl (Routing) durchzuführen.

Schicht 4: Transportschicht

Die Transportschicht hat die Aufgabe, dem Benutzer eine zuverlässige Ende-zu-Ende-Verbindung zur Verfügung zu stellen. Die angebotenen Dienste beinhalten hauptsächlich:

- Aufbau einer Transportverbindung,
- Datenübertragung,
- Verbindungsabbau.

Schicht 5: Sitzungsschicht

Die Hauptaufgabe der Sitzungsschicht ist die Organisation eines geregelten Ablaufs von Programmen, die in verschiedenen Rechnern laufen und dabei miteinander kommunizieren. Ein Benutzer kann sich z.B. in einer Sitzung an einem entfernten System anmelden oder Dateien zwischen Maschinen übertragen. Typische Aufgaben sind u.a. Sitzungsaufbau und –abbau, Synchronisation und Kontrolle des Sitzungsstatus.

Schicht 6: Darstellungsschicht

In der Regel sprechen verschiedene Systeme bei einem Datenaustausch zunächst unterschiedliche Sprachen. Die Darstellungsschicht stellt eine für das ganze System einheitliche Sprache mit einer abstrakten Syntax zur Verständigung zwischen Teilnehmern unterschiedlicher Hersteller zur Verfügung.

Schicht 7: Anwendungsschicht

Die Anwendungsschicht stellt die Verbindung zwischen dem Anwender und dem Netzwerk her. Analog der Benutzerschnittstelle eines komfortablen Betriebssystems mit seinen verschiedenen Diensten bietet die Anwenderebene des ISO-Modells Dienste für den Umgang eines Benutzers mit einem Netzsystem. Dazu gehören z.B. Zugriffe und Manipulationen in verteilten Datenbanken, Beeinflussung von Aufträgen in entfernten Rechnern, rechnergestützte Nachrichtensysteme, Grafikdienste und benutzerspezifische Anwendungen.

3.2.2 Modifiziertes Schichtenmodell bei Echtzeitverhalten

In industriellen Anwendungen werden sehr hohe Anforderungen an das Echtzeitverhalten der Automatisierungsfunktionen gestellt, d.h. man erwartet auf ein Ereignis hin innerhalb einer garantierten Zeit eine entsprechende Reaktion. Diese Anforderung muß nun, wenn Automatisierungsfunktionen über ein Netzwerk global verknüpft sind, auch auf die Kommunikationsmechanismen ausgedehnt werden. Mit wachsender Anzahl von Schichten wird der Funktionsumfang des Kommunikationsmodells erweitert. Aber es wird dabei auch eine längere Rechenzeit beansprucht. Die Folge ist eine längere Reaktionszeit. Für die Übertragung eines zeitkritischen Signals ist die Zeitvorgabe (Reaktionszeit < 10ms) anhand des ISO/OSI Referenzmodell nicht zu erreichen. Die Überlegungen entwickeln sich entsprechend dahingehend, eine reduzierte Architektur zu verwenden, wobei die unnötigen Schichten bzw. Funktionen entfernt werden. Was man dafür zusätzlich braucht, ist die Implementierung einer geeigneten Schnittstelle zwischen den Schichten, damit die untere Schicht den Dienst anbieten bzw. die obere Schicht den Dienst in Anspruch nehmen kann.

Für eine ausreichende und sichere Verständigung sind die Schichten 1, 2 und 7 unbedingt erforderlich. Schicht 1 legt die physikalischen Bedingungen wie z.B. Strom und Spannungspegel fest. In Schicht 2 wird der Zugriffsmechanismus und die Adressierung des Teilnehmers definiert. Dadurch kann zu einer bestimmten Zeit nur ein Teilnehmer Daten über den Bus senden. Die Anwendungsschicht 7 bietet ein notwendiges Interface für den Anwender. Abbildung 3.3 zeigt das modifizierte Schichtenmodell für eine Simatic S7, die ein Echtzeitverhalten erfüllen soll.

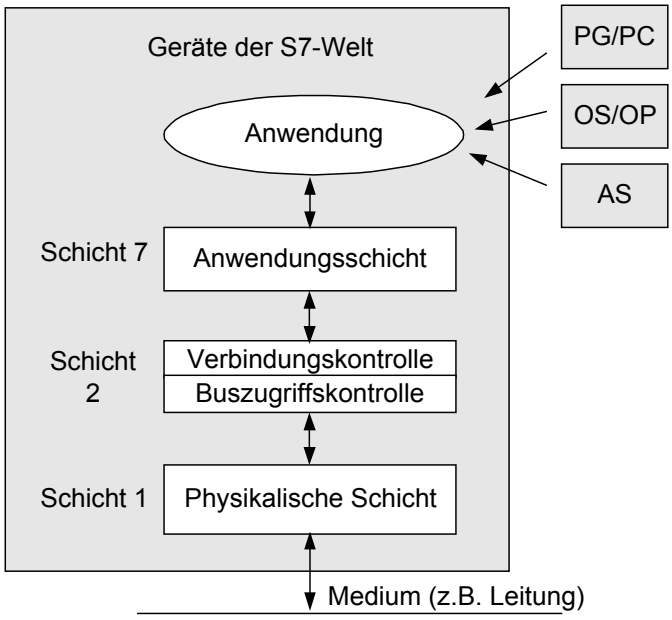


Abbildung 3.3: Referenzmodell in den zeitkritischen Fällen

3.3 Verbindungen zwischen den Kommunikationspartnern

Eine Verbindung ist eine logische Zuordnung zweier Kommunikationspartner zur Ausführung von Kommunikationsdiensten (Abbildung 3.4). Die Verbindung ist direkt mit einem Kommunikationsdienst verknüpft. Jede Verbindung hat zwei Endpunkte (jeweils auf der betreffenden CPU bzw. CP), welche die notwendigen Informationen zur Adressierung des Kommunikationspartners sowie weitere Attribute für den Verbindungsaufbau enthalten. Die Kommunikationsfunktionen im Anwenderprogramm stellen lediglich den lokalen Endpunkt der Verbindung dar.

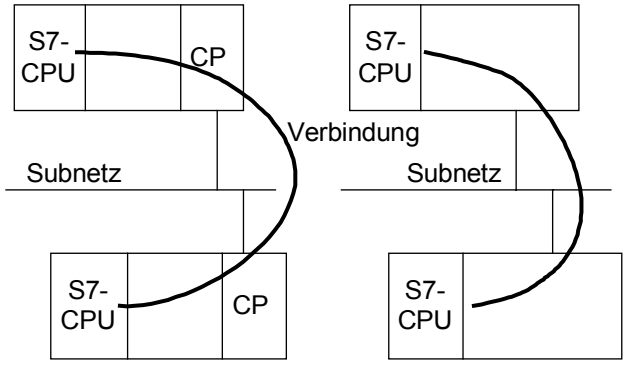


Abbildung 3.4: Verbindungskonfiguration

In der SIMATIC S7-Familie werden Verbindungen bezüglich Verbindungsart, Verbindungsauf-/-abbau und Verbindungstyp wie folgt klassifiziert:

Verbindungsart

In Abhängigkeit von der eingesetzten Software-Schnittstelle verlangen die zugehörigen Kommunikationsfunktionen entweder projektierte oder nichtprojektierte Verbindungen.

Projektierte Verbindungen

Diese Verbindungsart wird über die Verbindungstabelle projektiert. Hierbei wird für den jeweiligen Verbindungsendpunkt eine lokale Identifikationsnummer (ID) vergeben. Diese lokale ID wird für die Parametrierung der Kommunikationsfunktionen benötigt.

Nichtprojektierte Verbindungen

Nichtprojektierte Verbindungen werden in STEP 7 grundsätzlich nicht über die Verbindungstabelle projektiert. Diese Verbindungen werden beim Aufruf der Kommunikationsfunktion implizit aufgebaut und nach Beendigung der Datenübertragung ggf. wieder abgebaut.

Verbindungsauf-/-abbau

Um einen geregelten Verbindungsaufbau zu gewährleisten, müssen die Verbindungspartner in einem Endpunkt aktiv und im anderen Endpunkt passiv sein. Anderenfalls kann die Verbindung nicht aufgebaut werden.

Jede CPU hat eine begrenzte Anzahl von Verbindungsressourcen zur Verfügung. In einer großen Anlage besteht die Möglichkeit, daß ein Engpaß bei Verbindungsressourcen entsteht. Um diese Problematik zu berücksichtigen, gibt es zwei Methoden für den Verbindungsaufbau, eine statische oder eine dynamische Verbindung:

Statische Verbindungen

Statische Verbindungen werden verwendet, wenn in einer Anlagenprojektierung genügend viele Verbindungsressourcen in den einzelnen Stationen vorhanden sind und nicht mehr freigegeben werden müssen. Dabei braucht der zeitkritische Verbindungsaufbau und -abbau bei der Planung nicht berücksichtigt zu werden, denn statische Verbindungen werden einmalig aufgebaut und bleiben dauerhaft bestehen.

Dynamische Verbindungen

Dynamische Verbindungen werden dazu verwendet, um nacheinander Daten mit verschiedenen Kommunikationspartnern auszutauschen bzw. die vorhandenen Verbindungsressourcen effektiver zu nutzen. Der tatsächliche Verbindungsaufbau und -abbau erfolgt nicht beim Anlauf der Station, sondern bei Bedarf durch eine explizite Anforderung aus dem Anwenderprogramm. Die Dauer des Verbindungsaufbaus und -abbaus muß deshalb bei zeitkritischen Prozessen unbedingt berücksichtigt werden.

Verbindungstypen

Die Verbindung stellt den Zugang von der Software-Schnittstelle zum Kommunikationsdienst her. Eine Verbindung ist direkt mit einem Kommunikationsdienst verknüpft. Deshalb gibt es je Kommunikationsdienst einen entsprechenden Verbindungstyp.

In SIMATIC gibt es hauptsächlich folgende Verbindungstypen:

Eine *S7-Verbindung* ist in allen S7-Geräten projektierbar. Sie kann auf allen Subnetz-Typen (MPI, PROFIBUS, Industrial Ethernet) eingesetzt werden. Eine S7-Verbindung entspricht der Schicht 7 des ISO-Referenzmodells.

Eine *ISO-Transportverbindung* ist nur geeignet für das Industrial Ethernet. ISO-Transportdienst entspricht der Schicht 4 des ISO-Referenzmodells. Sie ermöglicht die Kommunikation zu einem Partner (z. B. SIMATIC S5 oder PC), der das Senden bzw. Empfangen von Daten gemäß ISO-Transport unterstützt.

Eine *FDL-Verbindung* (Fieldbus Data Link) unterliegt der Norm EN 50170 Vol.2 PROFIBUS und ist nur geeignet für ein PROFIBUS-Subnetz. Sie sorgt für die Übertragung von Daten zu

einem Kommunikationspartner (z. B. SIMATIC S5 oder PC), der das Senden bzw. Empfangen entsprechend der SDA-Funktion (Send Data with Acknowledge) unterstützt. Der FDL-Transportdienst entspricht der Schicht 2 des ISO-Referenzmodells.

Eine *FMS-Verbindung* (Fieldbus Message Specification) entspricht der europäischen Norm EN 50170 Vol.2 PROFIBUS. Sie ist mehr für die Übertragung von strukturierten Daten (FMS-Variablen) am PROFIBUS gedacht. Die FMS-Verbindung läßt sich in die Schicht 7 des ISO-Referenzmodells einordnen.

Eine *TCP-Verbindung* entspricht dem Standard TCP/IP (Transmission Control Protocol/Internet Protocol) und ermöglicht Kommunikation zu einem Partner (z. B. PC oder Fremdsystem), der das Senden bzw. Empfangen von Daten gemäß ISO-on-TCP unterstützt. Sie gilt nur für das Industrial Ethernet.

Eine *UDP-Verbindung* ist nur für das Industrial Ethernet vorgesehen und ermöglicht eine ungesicherte Übertragung zusammenhängender Datenblöcke zwischen zwei Teilnehmern.

3.4 Kommunikationsdienst

Bei der programmgesteuerten Kommunikation kann der Anwender durch Aufruf einer Kommunikationsfunktion im Anwenderprogramm die gewünschte Funktionalität, d.h. den Zeitpunkt, die Datenmenge und das Übertragungsverfahren, explizit bestimmen.

Für den Datenaustausch in der SIMATIC stehen auf der S7-300/400 und C7-600 entsprechende Kommunikationsfunktionen (SFCs, SFBs, FC/FBs) zur Verfügung.

SFCs für die S7-Basis-Kommunikation

Kommunikations-SFCs bieten die Möglichkeit einer quittierten Datenübertragung über nichtprojektierte S7-Verbindungen. Mit diesen Kommunikationsfunktionen können alle Kommunikationspartner am MPI-Subnetz erreicht werden. Die Verbindungen zu den Kommunikationspartnern werden bei Aufruf des SFCs dynamisch aufgebaut. Hierfür wird in den Kommunikationspartnern jeweils eine freie Verbindungsressource benötigt. Mit diesen Funktionen können kleine Datenmengen (max. 76 Bytes) über das MPI-Subnetz oder innerhalb einer S7 Station übertragen werden.

Die stellvertretende SFCs sind XSEND/XRCV für zweiseitige Kommunikation, wo bei beiden Kommunikationsteilnehmern projektiert werden muß, und X_GET/X_PUT für einseitige Kommunikation, wo nur bei einem Teilnehmer projektiert wird und bei dem anderen die Kommunikation durch das Betriebsystem abgewickelt wird.

Die Kommunikations-SFCs können auf allen S7-200/300/400- und C7-600-CPUs eingesetzt werden und dienen dem Datenaustausch mit S7/M7-300/400- und C7-600-CPUs.

SFBs für die S7-Kommunikation

Mit diesen SFB-Funktionen können Daten bis max. 64 kBytes über die Subnetze MPI, PROFIBUS und Industrial Ethernet übertragen werden. Es können nicht nur Daten übertragen, sondern auch weitere Kommunikationsfunktionen zum Steuern und Überwachen des Kommunikationspartners verwendet werden.

Die Kommunikations-SFBs (BSEND/BRCV) bieten die Möglichkeit einer quittierten Datenübertragung über projektierte S7-Verbindungen. Diese Verbindungen werden mit STEP 7 eingerichtet. Mit den PUT/GET-Funktionen können von einer S7-400 auch Daten von einer S7-300/C7-600 gelesen bzw. geschrieben werden.

Diese Kommunikations-SFBs können auf allen S7-400- und C7-600-CPUs eingesetzt werden. Sie dienen dem Datenaustausch mit S7/M7-300/400-CPUs.

FCs für S5-kompatible Kommunikation

Es gibt auch FCs, die sowohl der Kommunikation zwischen SIMATIC S7 als auch der Kommunikation von der SIMATIC S7 zur SIMATIC S5 sowie zu Nicht-S7-Stationen (z.B. PC) dienen. Sie ermöglichen einen einfachen Datenaustausch zwischen zwei

Kommunikationspartnern ohne Quittierung auf Anwenderebene über eine mit STEP7 projektierte Verbindung.

Diese Schnittstelle bilden die ladbaren Bausteine AG_SEND (AG_LSEND) und AG_RECV (AG_LRECV) bei S7 oder die Hantierungsbausteine SEND und RECEIVE bei S5. Für die S5-kompatible Kommunikation stehen beim Industrial Ethernet die Dienste ISO-Transport, ISO-on-TCP, und UDP, beim PROFIBUS der FDL-Dienst zur Verfügung.

Die FETCH/WRITE-Schnittstelle dient in erster Linie dem Anschluß der SIMATIC S7 an die SIMATIC S5 sowie weiteren Nicht-S7-Stationen (z.B. PC). Damit kann eine für SIMATIC S5 erstellte Software unverändert weiter genutzt werden. Sie ermöglicht den Datenaustausch über das Industrial Ethernet (ISO-Transport, ISO-on-TCP und TCP). Nur der Verbindungspartner (SIMATIC S5 oder Nicht-S7-Station) kann lesend (FETCH) oder schreibend (WRITE) auf Systemdaten in der SIMATIC S7 zugreifen.

FBs für die Standard-Kommunikation (FMS)

Die FMS-Schnittstelle (offene Kommunikation auf Schicht 7 nach dem ISO-Referenzmodell gemäß PROFIBUS-Norm) dient in erster Linie dem Datenaustausch mit Fremdsystemen ohne Quittierung auf Anwenderebene über eine mit STEP7 projektierte Verbindung über PROFIBUS. Es können Daten bis zu max. 237 Bytes übertragen werden. Außerdem umfassen die FMS-Dienste Variablendienste für strukturierte Daten (Variablen) und Verwaltungsdienste.

Der besondere Nutzen des FMS-Dienstes besteht darin, daß die Datenstrukturen in einer neutralen Form übertragen und im Kommunikationspartner umgesetzt werden. In den Anwenderprogrammen der Stationen verwendet der Anwender davon unberührt die jeweilige “Programmiersprache", z.B. AWL für SIMATIC S7 und C für die PC-Anwendungen.

SFBs für eine Punkt-zu-Punkt-Kopplung

Eine Punkt-zu-Punkt-Kopplung ermöglicht den Datenaustausch über eine serielle Verbindung bis max. 4 kBytes mit mittlerer Geschwindigkeit. Eine mit STEP 7 projektierte Punkt-zu-Punkt-Verbindung wird dabei benötigt.

Mit dem Punkt-zu-Punkt-CP für die S7-300/400 bzw. C7-600 können alle Kommunikationspartner gekoppelt werden, welche die Prozedur 3964(R), RK512 oder ASCII beherrschen. Fremdprotokolle werden über den ladbaren Treiber realisiert.

3.5 Netztypen

In jeder Automatisierungsebene (Leit-, Zellen-, Feld- und Aktor-Sensor-Ebene) wird eine spezifische Anforderung an das Netz gestellt. Den unterschiedlichen Anforderungen entsprechend werden auch unterschiedliche Netztypen angeboten:

- Multi Point Interface,
- PROFIBUS,
- Industrial Ethernet,
- Punkt-zu-Punkt-Kopplung,
- A/S-Interface.

3.5.1 Multi Point Interface

Jede Zentralbaugruppe bei SIMATIC S7 hat eine "mehrpunktfähige Schnittstelle" (MPI, Multi Point Interface). Sie ermöglicht den Aufbau eines Subnetzes, in dem Automatisierungssystem, Bedien- und Beobachtungsgeräte und Programmiergeräte untereinander Daten austauschen können, und ist geeignet vor allem für die Feld- und Zellenebene mit kleiner Ausdehnung und kleiner Teilnehmerzahl. Der Datenaustausch wird über ein Siemens-eigenes Protokoll abgewickelt. Die technischen Daten sind der Tabelle 3.1 zu entnehmen.

Normung	SIEMENS spezifisch
Stationen	Maximal 32
Zugriffsverfahren	Token
Übertragungsrate	19,2 k Bit/s, 187,5 kBit/s oder 12 MBit/s
Übertragungsmedium	Geschirmte Zweidrahtleitung, LWL (Glas oder Kunststoff)
Netzausdehnung	Segmentlänge 50 m, über RS 485 Repeater bis 1100 m, mit LWL über OLM > 100 km
Topologie	Elektrisch: Linie Optisch: Baum, Stern, Ring

Tabelle 3.1: Technische Daten des MPI

3.5.2 PROFIBUS

PROFIBUS steht für "Process Field Bus" und ist ein herstellerunabhängiger Standard nach EN50170 Volume 2 für die Vernetzung im Zell- und Feldbereich. Er wird eingesetzt zur Übertragung kleinerer bis mittlere Datenmengen.

Der PROFIBUS wird in zwei Ausprägungen angeboten:

- als Feldbus PROFIBUS DP für schnellen, zyklischen Datenaustausch und PROFIBUS PA für den eigensicheren Bereich,
- im Zellbereich als PROFIBUS für die schnelle Übertragung mit gleichberechtigten Kommunikationspartnern.

Physikalisch ist der PROFIBUS ein elektrisches Netz auf Basis einer geschirmten Zweidrahtleitung oder ein optisches Netz auf Basis eines Lichtwellenleiters (LWL) oder drahtlos mit Infrarottechnik.

Buszugriffsverfahren

Sind an einem Kommunikationsnetzwerk mehrere Master und Slaves konfiguriert, wird das Token Bus Verfahren benutzt. Die Berechtigung für den Zugriff auf den Bus muß in diesem Fall geteilt werden.

Besteht ein logischer Ring aus nur einem aktiven Teilnehmer und befinden sich am Bus mehrere passive Teilnehmer, so entspricht dies einem reinen Master-Slave-System. Die typische PROFIBUS-DP-Buskonfiguration basiert auf diesem Buszugriffsverfahren. Ein aktiver Teilnehmer (DP-Master) tauscht in zyklischer Reihenfolge Daten mit den passiven Teilnehmern (DP-Slaves) aus. Das Zugriffsverfahren erlaubt das Aufnehmen bzw. Entfernen von Stationen während des Betriebs. Der Netzzugriff beim PROFIBUS entspricht der in der EN 50170, Volume 2, festgelegten Methode des "Token Bus Verfahren" für aktive und des "Master-Slave" für passive Stationen (Abbildung 3.5).

Enthält ein Netzwerk beide Komponenten, so werden auch beide Zugriffsverfahren zum Einsatz kommen. Der Token wird innerhalb der aktiven Teilnehmer in Umlauf gebracht. Das Master-Slave-Verfahren ermöglicht dem Master (aktiver Teilnehmer), der gerade die Sendeberechtigung (Token) besitzt, die ihm zugeordneten Slaves (passive Teilnehmer) anzusprechen. Der Master hat hierbei die Möglichkeit, Nachrichten an die Slaves zu übermitteln bzw. von den Slaves abzuholen. Nach Beendigung dieser Arbeit oder nach einer bestimmten Zeit wird der Token an den folgenden Master weitergegeben. Die technischen Daten sind der Tabelle 3.2 zu entnehmen.

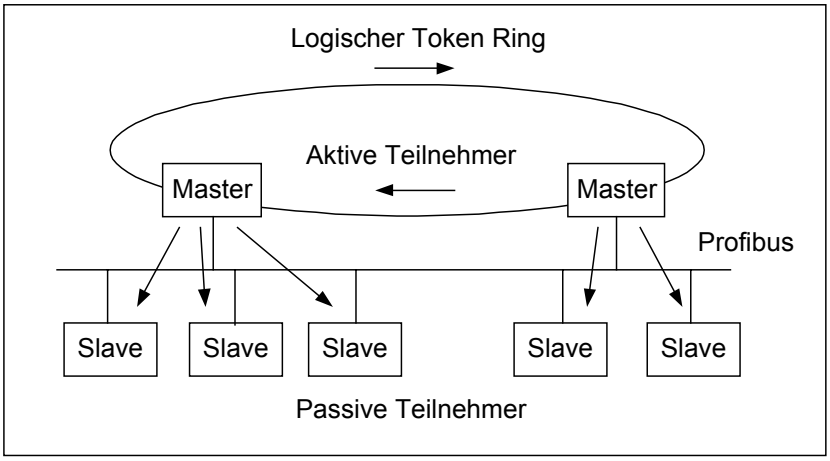


Abbildung 3.5: Funktionsprinzip des PROFIBUS-Zugriffsverfahrens

Normung	EN 50170 Volume 2 PROFIBUS
Stationen	Maximal 127 Stationen im Netz
Zugriffsverfahren	- Token Bus für die Buszuteilung unter aktiven Stationen. - Master-Slave für die Kommunikation mit passiven Stationen.
Übertragungsrate	9,6 kBit/s-12 MBit/s
Übertragungsmedium	Geschirmte Zweidrahtleitung oder Lichtwellenleiter
Netzausdehnung	Segmentlänge 1000 m, über Repeater bis 10 km, mit LWL über OLM > 100 km
Topologie	Linie, Baum, Stern, Ring

Tabelle 3.2: Technische Daten des PROFIBUS

3.5.3 Industrial Ethernet

Das Industrial Ethernet ist das Subnetz für den Verbund von Rechnern und Automatisierungssystemen mit Einsatzschwerpunkt im industriellen Bereich, definiert durch den internationalen Standard IEEE 802.3. Es dient dem Austausch großer Datenmengen und kann für die Übertragung über große Entfernungen eingesetzt werden. Physikalisch ist Ethernet ein elektrisches Netz auf der Basis einer geschirmten Koaxialleitung, einer Twisted Pair Verkabelung oder eines optischen Netzes auf der Basis eines Lichtwellenleiters.

Buszugriffsverfahren

Das Zugriffsverfahren ist das stochastische (zufallsabhängige) Zugriffsverfahren CSMA/CD (Carrier Sense Multiple Access with Collision Detection, genormt in IEEE 802.3). Dabei darf jeder Teilnehmer zu beliebigen Zeitpunkten senden, vorausgesetzt, auf dem Bus wird von einem anderen Teilnehmer zur Zeit nicht gesendet. Zu Konflikten kommt es aufgrund von Signallaufzeiten, wenn zwei Teilnehmer bei freiem Bus gleichzeitig mit dem Senden beginnen. In diesem Fall erkennen beide Teilnehmer durch Mithören auf dem Bus die Kollision, hören mit dem Senden auf und beginnen einen neuen Sendeversuch nach einer stochastischen Wartezeit. Im ungünstigen Fall erkennt die Empfängerseite die Kollision sofort, aber die Sendeseite erst nach doppelten Übertragungszeiten ts. Daraus läßt sich ableiten, daß die minimale Sendedauer eines Pakets 2*ts sein muß, um eine sichere Kollisionserkennung zu gewährleisten. In der Praxis bedeutet dies, daß die minimale Paketlänge sowohl von der Datenübertragungsrate als auch von der Leitungslänge abhängt. Busse mit CSMA/CD-Verfahren wie z.B. Industrial Ethernet haben meist eine Übertragungsrate (ÜR) von 10MBit/s,.

Beispiel: Bei l = 1000 m ergibt sich eine Signallaufzeit ts = 5 us (v = 0,66 * c)

Wird mit einer ÜR = 1 MBd gearbeitet, muß die Information mindestens 10 Bit lang sein, damit eine Kollision sicher erkannt werden kann.

Die technischen Daten sind der Tabelle 3.3 zu entnehmen.

Normung	IEEE 802.3
Stationen	mehr als 1.000
Zugriffsverfahren	CSMA/CD
Übertragungsrate	100 MBit/s
Übertragungsmedium	elektrisch: 2-fach geschirmte Koaxialleitung Industrial Twisted Pair optisch: Lichtwellenleiter
Netzausdehnung	elektrisch: 1,5 km optisch: 4,5 km
Topologie	Linie, Baum, Stern, Ring

Tabelle 3.3: Technische Daten des Industrial Ethernet

3.5.4 Punkt-zu-Punkt-Kopplung

Eine Punkt-zu-Punkt-Kopplung ist kein Subnetz im herkömmlichen Sinne. In der SIMATIC wird diese Kopplung mittels Punkt-zu-Punkt-Kommunikationsprozessoren (CP) realisiert, wobei zwei Stationen miteinander verbunden sind.

Eine Punkt-zu-Punkt-Kopplung ermöglicht den Datenaustausch über eine serielle Schnittstelle zwischen

- Automatisierungsgeräten,

- Automatisierungsgeräten und PG/PC,
- Automatisierungsgeräten und Fremdsystemen.

Die Vorteile der Punkt-zu-Punkt-Kopplung bestehen in folgendem:

- gute Anpassungsmöglichkeit an die Prozedur des Kommunikationspartners mit Hilfe von Standardprozeduren oder nachladbaren Treibern,
- Definition einer eigenen Prozedur über ASCII-Zeichen.

3.5.5 A/S-Interface

Das A/S-Interface oder Aktor-/Sensor-Interface ist ein Subnetzsystem für die unterste Prozeßebene in Automatisierungsanlagen. Es dient speziell zur Vernetzung binärer Sensoren und Aktoren. Die Datenmenge beträgt maximal 4 Bits pro Slave-Station.

Das A/S-Interface ist ein sogenanntes "Single-Master-System", d.h. es gibt pro A/S-Interface-Subnetz nur einen Master, der den Datenaustausch steuert. Er ruft nacheinander alle Slaves auf und liest oder schreibt die Daten. Der Master-Slave-Zugriff mit zyklischem Polling gewährleistet eine definierte Reaktionszeit.

Die wichtigsten Eigenschaften sind im folgenden aufgelistet:

- Das A/S-Interface ist optimiert für den Anschluß binärer Sensoren und Aktoren. Über die A/S-Interface-Bus erfolgt sowohl der Datenaustausch zwischen Sensoren/Aktoren und dem Master als auch die Stromversorgung der Sensoren.
- Der A/S-Interface-Master benötigt für den zyklischen Datenaustausch mit bis zu 31 Stationen maximal 5 ms.
- Eine Vielzahl von Geräten (Aktoren/Sensoren) ist durch die herstellerunabhängige Normung anschließbar.

3.6 Kopplung von Bussystemen

Um einen durchgängigen Informationsfluß zwischen zwei unterschiedlichen Subnetzen zu gewährleisten, sind spezielle Koppelelemente erforderlich. Im Normalfall sind die zu koppelnden Subnetze historisch gewachsen. Die Subnetze können nicht ohne weiteres verbunden werden, da ankommende Informationen aus Subnetz A von den Protokollen des Subnetzes B nicht interpretiert werden können. Eine wesentliche Forderung ist hierbei, daß sich die gekoppelten Subnetze aus Benutzersicht wie ein einziges Subnetz verhalten, d.h. daß sich durch Kopplung keine Einbußen in der Funktionalität ergeben dürfen. Die Kopplung ist damit für die Benutzer transparent. Transparenz bedeutet in diesem Zusammenhang “Unsichtbarkeit“, d.h. man will Subnetze so koppeln, daß der Benutzer es möglichst nicht merkt und er die Software nicht ändern muß.

Je nach Aufwand der Kopplung bzw. der Unterschiedlichkeit der zu koppelnden Subnetze kann zwischen Repeater, Bridge, Router und Gateway zur Netzwerkkopplung unterschieden werden /Tan97/. Diese Koppelelemente lassen sich aufgrund ihrer Aufgaben auf das ISO-Referenzmodell abbilden (Abbildung 3.6).

Repeater

Die Anzahl der Busteilnehmer und Buslänge sind durch physikalische Gegebenheiten und logische Festlegung beschränkt. Um diese Restriktionen zu überwinden, gibt es häufig die Möglichkeiten, gleichartige Busse miteinander zu verbinden. Die Verbindung zwischen zwei Segmenten wird durch den Repeater hergestellt, der beidseitig an einem Buskoppler angeschlossen ist. Aufgabe des Repeaters ist es, die auf der Leitung empfangene Information auf die jeweils andere Seite zu übertragen und zu verstärken.

Ein Repeater wirkt für alle Schichten der kommunizierenden Teilnehmer transparent, d.h. bereits die physikalischen Schichten beider Netzwerke müssen identisch sein. Ein Repeater kann sowohl zur Kopplung zweier gleichartiger Subnetze als auch zur Vergrößerung bzw. Verlängerung eines bestehenden Subnetzes, z.B. eines Bussystems, eingesetzt werden.

Bridge

Bridge wird zur Verbindung von Subnetzen eingesetzt, die in der Sicherungsschicht (Logical Link Control LLC) mit denselben Protokollen arbeiten. Die Übertragungsmedien und die Buszugriffsverfahren (Medium Access Control, MAC) der zu verbindenden Subnetze können unterschiedlich sein. Bridge wird hauptsächlich dann eingesetzt, wenn lokale Netzwerke unterschiedlicher Topologien zu verbinden sind oder wenn durch spezielle Anwendungen bestimmte Strukturen an Subnetze angebunden werden sollen. Die Aufgaben des Bridge beziehen sich bei manchen Ausführungen nur auf den Buszugriff (MAC). Die LLC bleibt hiervon unberührt. Diese Art von Bridge kommt bei Subnetzen zum Einsatz, die lediglich ein unterschiedliches Übertragungsmedium (z.B. Zweidrahtleitung, Lichtwellenleiter) besitzen, ansonsten aber identisch aufgebaut sind.

Router

Der Router dient zur Verbindung von Netzwerken, die unterschiedliche Protokolle in der physikalischen Schicht und Sicherungsschicht haben. Der Router bestimmt zusätzlich den optimalen Weg einer Nachricht durch ein bestehendes Netzwerk. Kriterien für den optimalen Weg können hierbei beispielsweise die Weglänge oder die geringste Übertragungsverzögerung sein. Um seine Aufgabe zu erfüllen, ändert der Router die Ziel- und Quelladressen der Vermittlungsschicht der ankommenden Datenpakete, bevor er sie weiterleitet. Da der Router eine wesentlich komplexere Aufgabe erfüllen muß als Bridge, hat er eine geringere Arbeitsgeschwindigkeit.

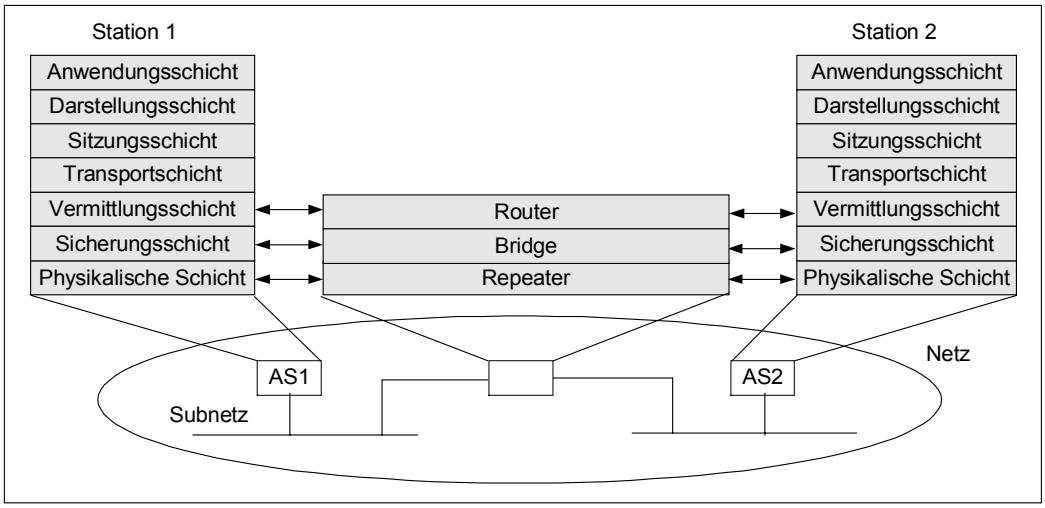


Abbildung 3.6: Repeater, Bridge und Router

Gateway

Ein Gateway dient zur Kopplung eines Bussystems mit anderen, strukturell sehr unterschiedlichen Kommunikationssystemen. Mit Hilfe von Gateway können zwei beliebige Subnetze verbunden werden.

Bezogen auf das ISO-Referenzmodell besteht die Aufgabe des Gateways darin, die ankommende Information von einem Netz in die für das andere gekoppelte Netz verständliche Information zu übersetzen. So müssen Regelwerke von den beiden gekoppelten Netzen (Protokoll) Gateway bekannt sein.

Ein Gateway ermöglicht auch die Kopplung eines ISO-Netzes, mit einem nicht ISO-Netz. Dann besitzt eine Hälfte in der Abbildung 3.7 nicht die 7-Schichten-Struktur sondern einen davon abweichenden Aufbau. Netzwerkverbindungen mittels Gateway sind i.a. durch einen erheblichen Aufwand und eine geringere Geschwindigkeit gekennzeichnet.

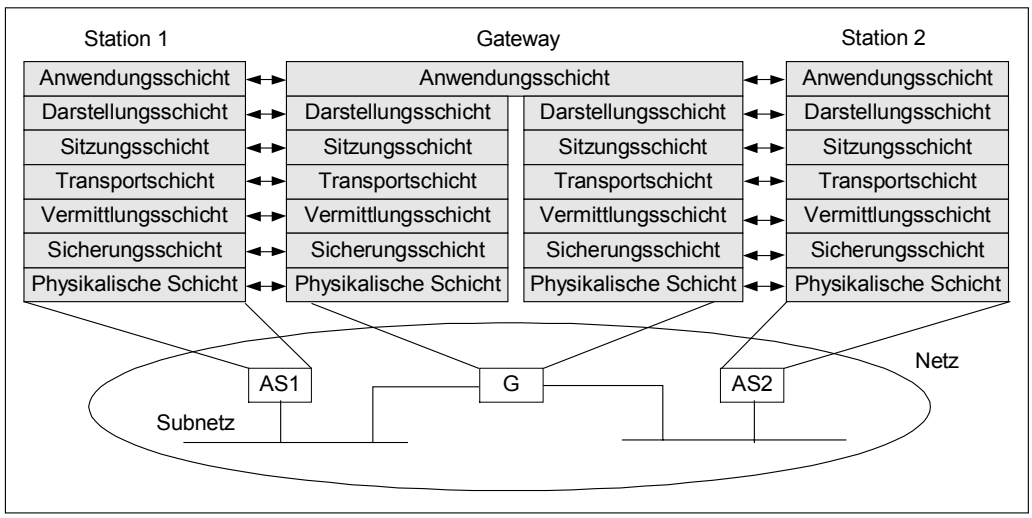


Abbildung 3.7: Gateway

3.7 Übertragungssicherheit

In der Schicht 1 findet eine physikalische Codierung der zu übertragenden Bits statt, um eine möglichst hohe Störsicherheit bzw. eine möglichst sichere Datenübertragung zu gewährleisten. Wenn Daten empfangen werden, sind diese oberhalb der Schicht 1 aufgrund der Störungen des Übertragungsmediums mit einer Fehlerwahrscheinlichkeit behaftet. Hierbei finden sich in der Literatur die Begriffe Bitfehlerrate und Blockfehlerwahrscheinlichkeit. In Schicht 2 findet eine Codierung zur Datensicherung statt. Eine Eigenschaft eines solchen Codes ist die sogenannte Hamming Distanz (HD). Sie gibt an, in wie vielen Bits sich zwei gültige Codewörter unterscheiden, d.h. wie viele Bits kippen müssen, damit wieder ein gültiges Codewort entsteht. Das Kippen von bis zu (HD-1) Bit wird also als Fehler erkannt.

Oberhalb der Schicht 2 bleibt schließlich eine Restfehlerwahrscheinlichkeit. Diese gibt das Verhältnis der unerkannt fehlerhaften Telegramme zur Gesamtanzahl der empfangenen Telegramme an. Daher ist die Restfehlerwahrscheinlichkeit als Maß für die Übertragungssicherheit zu sehen. Diese hängt ab von den Störungen auf der Leitung, der verwendeten physikalischen Codierung (z.B. NRZ, Manchester Codierung) und der nachrichtentechnischen Codierung (Telegramm).

Die Hamming Distanz ist daher nur bedingt als Kriterium zur Beurteilung der Übertragungssicherheit sinnvoll. Wenn man eine bestimmte Bitfehlerwahrscheinlichkeit und eine feste Hamming Distanz zugrunde legt, steigt mit der Telegrammgröße auch die Restfehlerrate. Wenn man bei der physikalischen Codierung einen entsprechenden Aufwand betreibt, erzielt man eine hohe Störsicherheit, so daß die Bitfehlerrate bzw. Blockfehlerwahrscheinlichkeit reduziert wird. Dies hat bei konstanter Hamming Distanz eine Reduktion der Restfehlerwahrscheinlichkeit zur Folge. Daher ist beim A/S-Interface-Bus trotz HD = 2 mit einer geringen Restfehlerwahrscheinlichkeit zu rechnen.

4 Technische Anforderungen und Meßmöglichkeiten

In diesem Kapitel werden zuerst die zu berücksichtigenden Punkte eines Kommunikationskonzepts für die industrielle Anwendung aufgestellt. Danach folgt ein Sitzungsmodell, an dem sich alle Konzepte anlehnen sollen, um eine spätere Integration in die gesamte Projektierungsumgebung zu ermöglichen. Zum Schluß werden die Meßgrößen zur Performanceanalyse eingeführt.

4.1 Technische Anforderungen an ein Kommunikationskonzept

Die Anforderung der Industrie nach immer leistungsfähigerer Kommunikation und immer niedrigeren Entwicklungskosten trotz höherer Komplexität der Automatisierungsanlagen zwingt dazu, die zur Zeit etablierten Mechanismen für die Kommunikation zu überdenken. Es soll untersucht werden, ob Methoden für die Beschaffung aktueller Prozeßwerte zur Anwendung kommen können, die bezüglich der folgenden Aspekte besser geeignet sind.

Reduktion des Datenaufkommens

Der heutige Ansatz, die gewünschten Daten zyklisch zu übertragen, birgt zwei grundsätzliche Nachteile: Zum einen wird durch die regelmäßige Anforderung der Daten das Kommunikationsmedium mit einer Fülle von Telegrammen belastet. Für kleine Datenmengen, wie sie bei der Kommunikation auf der Feldebene üblicherweise auftreten, werden mehrfach Telegramme gesendet. Zum anderen wird ohne Berücksichtigung der Qualität und des Ursprungs der interessierenden Größen deren Transfer mit einer meistens zu hohen Rate durchgeführt. Es ist deswegen sinnvoll, einen Mechanismus zu finden, der den Zeitpunkt des Datenaustauschs je nach der Kommunikationsanforderung flexibel gestalten kann. Dabei soll dadurch sowohl das Datenaufkommen als auch der Projektierungsaufwand gesenkt werden.

Technische Entkopplung der beteiligten Komponenten

Jedes Automatisierungssystem soll unabhängig von anderen Geräten in der Prozeßebene sein und kann separat in Betrieb genommen werden. Das Kommunikationskonzept soll diese Vorgehensweise nicht stören, sondern ebenfalls tolerant gegenüber Nach- und Teilprojektierung der Kommunikationspartner sein. Es soll möglichst wenig Wechselwirkung zwischen den Mechanismen der Kommunikation und Projektierung geben, womit ein separates Laden erst möglich wird.

Überwachungsmechanismen

Nach der Inbetriebnahme soll ein neues Automatisierungssystem von den bereits im Netz vorhandenen Automatisierungssystemen automatisch erkannt werden, die sowohl Nachbar-Automatisierungssysteme als auch Entfernte-Automatisierungssysteme sein können. Die Kommunikation dazwischen findet anschließend automatisch statt. Das gleiche soll auch für ein ausgefallenes Automatisierungssystem gelten. Dabei soll die Projektierungsarbeit von außen nach Möglichkeit vermindert werden.

Usability

Die Kundenorientierung spielt eine immer wichtiger werdende Rolle bei der Produktentwicklung. Die Anforderung der Kunden variiert je nach dem Anwendungsbereich. Um diesen Anforderungen möglichst gerecht zu werden, kommt dem Softwareanteil in Zukunft eine immer größere Bedeutung innerhalb einer Automatisierungslösung angesichts ihrer kürzeren Entwicklungszeit und geringeren Entwicklungskosten im Vergleich zur Hardwareentwicklung zu. Es bedeutet aber auch zugleich, daß mehr Softwareprojektierung notwendig ist. Um die Projektierungskosten zu reduzieren, muß bei der Konzeptaufstellung die *Usability* entsprechend berücksichtigt werden.

Die Kommunikation zwischen den Automatisierungssystemen wird mit Hilfe von Bausteinen projektiert. Es ist wünschenswert, die Kommunikation mit möglichst wenigen Bausteinen zu ermöglichen. Eigentlich handelt es sich bei einer Kommunikation um einen Dienst, der für den Anwender weitestgehend verdeckt ablaufen sollte. Die Einstellung von Kommunikationsparametern soll nach Möglichkeit systemintern vergeben werden, damit ein großer Teil der Projektierungsarbeit dem Anwender erspart bleibt.

Ein denkbarer Ansatz zur Erleichterung der Projektierung ist, die Projektierung der Kommunikation durch Verschaltung vorzunehmen. Genauer gesagt, beim Erstellen der Continuous Function Chart (CFC)/Sequential Function Chart (SFC) - Pläne werden ressourcenübergreifende Verschaltungen analog zu CPU-lokalen Verschaltungen projektiert (Abbildung 4.1). Mit dieser Vorgehensweise wird der Projektierungsaufwand erheblich reduziert, z.B bezüglich der externen Verwendung von Kommunikationsbausteinen und der Parametereinstellungen. Gleichzeitig werden aber mehr Rechenbelastungen auf den Automatisierungssystemen entstehen, die wiederum die Performance negativ beeinflußt.

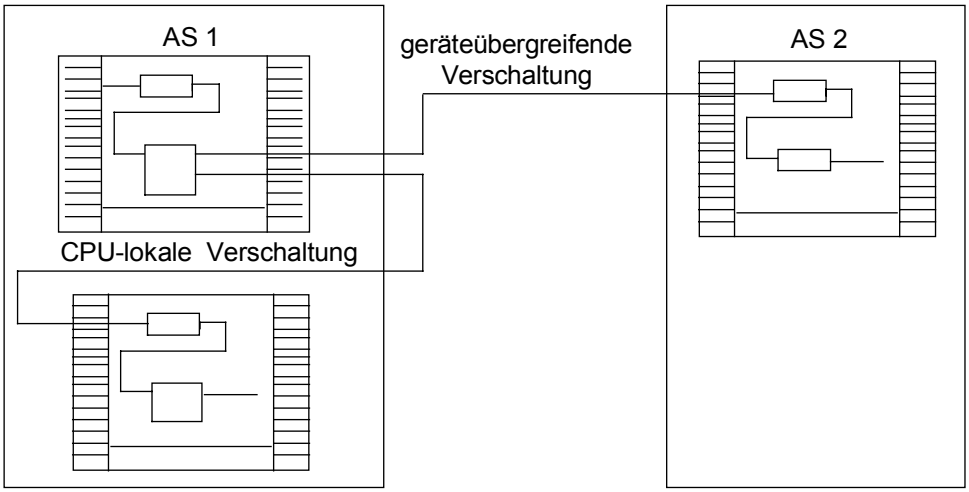


Abbildung 4.1: AS-AS Kommunikation durch geräteübergreifende Verschaltung

Änderungsprojektierung

Die Kriterien für eine Zugänglichkeit des Kommunikationskonzepts bei der Änderungsprojektierung ist, daß der Vorgang möglichst wenig Aufwand bei der Umsetzung und mit möglichst wenig Einfluß auf den Prozeß verbunden sein soll. Es sind folgende Punkte zu beachten:

- Bzgl. der Änderungen ist zu berücksichtigen, daß alle Änderungen (auch solche, die die Kommunikation betreffen) im Zustand RUN in die CPU geladen werden können.
- Die Ressourcen sollen abhängig vom Projektierungsinhalt neu belegt oder freigegeben werden.
- Bei Delta-Laden (Laden für die Änderungsprojektierung) sollen die bestehenden Verschaltungen weitgehend erhalten bleiben, damit der RUN-Betrieb nicht beeinträchtigt wird.

Kompatibilität

Als Voraussetzung für die Kompatibilität der neuen Version mit bestehenden Applikationen müssen die neuen Mechanismen als Erweiterung der bestehenden Lösung aufgefaßt und in diesem Sinne realisiert werden. Es darf keine Verkopplung zwischen alten und neuen Mechanismen vorgenommen werden, wenn die Funktionsweise der bestehenden Systeme bei der Realisierung keinesfalls negativ beeinflußt wird.

Das heißt, das Modell soll eine ausgedehnte, standardisierte Funktionalität bieten, die es erlaubt, durch die Aktivierung zusätzlicher Funktionen das in Zukunft notwendige Funktionsspektrum abdecken zu können und die bestehende Funktionalität nicht zu beeinflußen.

Alte Bausteine sollen neben neuen, die bereits auf das verbesserte Kommunikationskonzept umgestellt worden sind oder aus der neuen Baustein-Bibliothek entnommen sind, bestehen

können. Der Zeitrahmen und das Maß der Umstellung eines Gerätes können so flexibel gestaltet werden.

Einfaches Programmiermodell

Um die Softwarearbeit für die Modifikation zu vereinfachen, sollen möglichst wenige, eindeutige Schnittstellen für die einfache und anschauliche Einbindung der neuen Mechanismen bestehen.

4.2 Modell der Kommunikationsprojektierung in der Zellenebene

Bevor ein Datenaustausch zwischen den Automatisierungssystemen stattfinden kann, müssen bestimmte Vorarbeiten geleistet werden. Es geht vor allem um die Projektierung der Verbindung, die Konfiguration der Arbeitsumgebung sowie anschließend um die Übersetzung und den Ladevorgang. Im folgenden wird ein Sitzungsmodell vorgegeben, mit dem alle notwendige Schritte bei einer Kommunikationsprojektierung in der Zellenebene angegeben werden. Eine Sitzung enthält folgende 5 Schritte:

1. Technologisch projektieren mit Ressourcen

In diesem Schritt wird das Anwenderprogramm erstellt, und zwar mit CFC für eine kontinuierliche Steuerung und mit SFC für eine Ablaufsteuerung. Außerdem wird hier auch die Kommunikation zwischen CPUs durch den Anwender projektiert oder modifiziert.

2. Konfigurieren der Arbeitsumgebung

Bevor die Kommunikationsteilnehmer unter sich Daten austauschen können, muß das Automatisierungssystem durch Projektierungsdaten noch konfiguriert werden.

Die Projektierung kann sich in 2 Schritte aufgliedern:

- Projektierung der Hardware,
- Projektierung der Kommunikationsverbindung.

Hardwareprojektierung

Bevor die projektierten Programme überhaupt laufen können, muß ein Automatisierungssystem einzeln konfiguriert werden. Dabei soll die vorhandene Hardwareumgebung mit Hilfe von HW-Config (Hardware Konfigurationstool) konfiguriert werden.

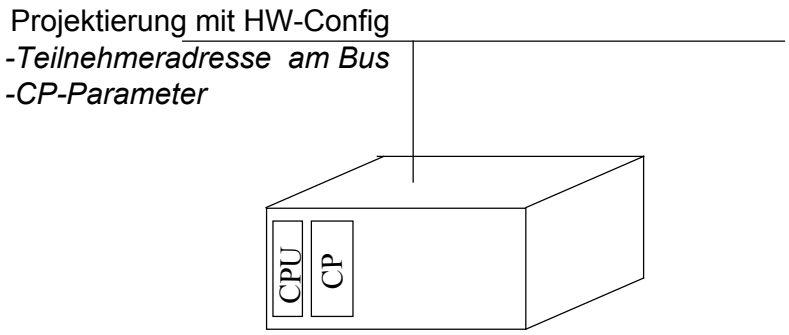


Abbildung 4.2: Hardwareprojektierung

Zu der Hardwareumgebung gehört auch die Netzprojektierung. Ein Netz in der industriellen Anwendung umfaßt alle Stationen, die miteinander Kommunikation betreiben wollen.

Ein Netz zu projektieren heißt:

- Einstellung der Netztypen:

  Je nach dem Anwendungszweck kann das Netz aus Profibus, Industrial Ethernet oder MPI bestehen. Der Netztyp muß in dem Anwenderprojekt bekannt sein.

- Einstellung der Netzeigenschaften (z.B. Übertragungsgeschwindigkeit):

  Die Übertragungsgeschwindigkeit zwischen CPUs kann unterschiedlich eingestellt werden (19.2 kbit/s, 187 kbit/s, 1.5 Mbit/s, 3 Mbit/s, 6 Mbit/s, 12 Mbit/s). Die Übertragungsgeschwindigkeit darf nicht größer sein als die, die der langsamste Teilnehmer unterstützt.

- logischer Anschluß der Kommunikationsteilnehmer an Netz:

  Ein Kommunikationsteilnehmer wird logisch an das Netz angeschlossen. Dabei werden die Teilnehmeradressen an den Netzen festgelegt. Die maximal zugelassene Adressenanzahl hängt von den Netztypen ab, wobei sie z.B bei MPI 32 beträgt.

Projektierung der Kommunikationsverbindung

Verbindungen, die zum Datenaustausch zwischen CPUs dienen, sind mit NetPro in SIMATIC STEP7 Basispaket zu projektieren. Eine Verbindung legt die logische Beziehung für jeweils zwei Kommunikationspartner fest. Kommunikationsverbindungen sind immer dann erforderlich, wenn im Anwenderprogramm Kommunikationsfunktionen durchgeführt werden sollen.

Eine Verbindung hat folgende Parameter und Eigenschaften:

- beteiligte Kommunikationspartner,
- Verbindungstyp (z.B. S7-Verbindung, FDL-Verbindung),
- andere spezielle Verbindungseigenschaften (z.B. aktiver/passiver Verbindungsaufbau).

Jede Verbindung benötigt auf den beteiligten Stationen Verbindungsressourcen für den Endpunkt bzw. für den Übergangspunkt. Die Anzahl der Verbindungsressourcen ist CPU-

spezifisch. So ist die zur Verfügung stehende Verbindungsanzahl auch begrenzt. Sind alle Verbindungsressourcen eines Kommunikationspartners belegt, so kann keine neue Verbindung aufgebaut werden.

3. Ressourcenzuordnung zur Hardware

Nach der Hardwareprojektierung werden die neutralen Ressourcen zur Hardware (CPU) zugeordnet, damit sie später richtig geladen werden können.

4. Codegenerierung

Übersetzt werden alle Pläne des aktuellen Planordners in das Programm, das anschließend in das Zielsystem geladen werden kann. Vor dem Übersetzen wird eine Konsistenzprüfung durchgeführt.

5. Laden

Der Code ist ablauffähig und kann geladen werden. Beim Laden gibt es zwei Möglichkeiten, entweder das Gesamt-Laden oder Delta-Laden.

Gesamt-Laden

Beim Gesamt-Laden werden nur die Code- und Datenbausteine vom Programmiergerät (PG) in den Lade- und den Arbeitsspeicher der CPU geladen. Die symbolische Operandenzuordnung (Symboltabelle) und die Bausteinkommentare bleiben im Speicherbereich des PG. Bevor dieser Vorgang stattfindet, muß das gegebenenfalls laufende Programm unterbrochen werden. Das heißt, die CPU muß in den Betriebszustand STOP gebracht werden. Das Laden kann sowohl über ein direkt an die CPU angeschlossenes Programmiergerät oder z.B. über einen PROFIBUS geschehen.

Um eine schnelle Bearbeitung des Anwenderprogramms zu gewährleisten und den nicht erweiterbaren Arbeitsspeicher nicht unnötig zu belasten, werden nur die Teile der Bausteine, die für die Programmbearbeitung relevant sind, in den Arbeitsspeicher geladen.

Bausteinteile, die nicht erforderlich sind, um das Programm ablaufen zu lassen (z. B. Bausteinköpfe), bleiben im Ladespeicher.

Abbildung 4.3 erläutert das Laden des Programms in den CPU-Speicher.

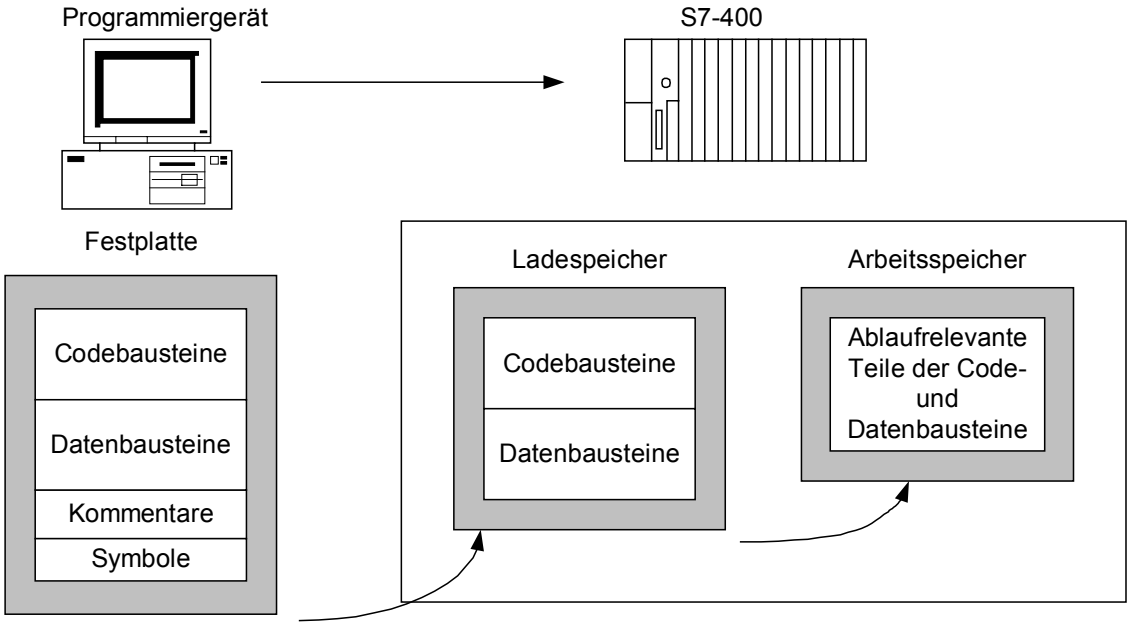


Abbildung 4.3: Ladevorgang

Delta-Laden

Beim Delta-Laden wird nur die Änderung gegenüber dem letzten Ladevorgang geladen. Die unveränderten Anteile bleiben auf der CPU stehen. Die CPU muß nicht in den Betriebszustand STOP gesetzt werden. Es ist besonders nützlich, wenn die Änderung ohne Unterbrechung der CPU geladen werden kann.

4.3 Meßmöglichkeiten bezüglich der Performanceanalyse

Die Performanceanalyse verfolgt das Ziel, den Einfluß der Systemparameter auf die Veränderung der vorher festgelegten Leistungskenngrößen zu bestimmen. Diese Parameter werden als die Einflußgrößen des Systems genannt (Abbildung 4.4). Folgend werden die untersuchten Leistungskenngrößen und Einflußgrößen aufgelistet.

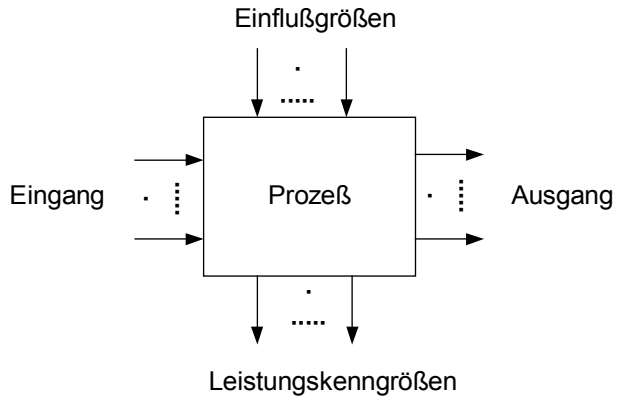


Abbildung 4.4: Zusammenhang zwischen Einflußgrößen und Leistungskenngrößen

Leistungskenngröße

Als Leistungskenngröße werden hier Übertragungszeit, Varianz und Fehlerquote /Nol94/ so definiert.

Übertragungszeit

Um die Bearbeitungszeit auf Sender und Empfänger auch zu berücksichtigen, wird die Übertragungszeit in dieser Arbeit so definiert:

Die Übertragungszeit ist die Zeit, in der der Prozeßwert vom Sender-Automatisierungssystem gesendet, von Empfänger-Automatisierungssystem empfangen und zurückgesendet, wieder vom Sender-Automatisierungssystem empfangen wird (Abbildung 4.5). Diese Zeit hängt von der Bearbeitungszeit der eingesetzten Kommunikationsmechanismen und der Transportzeit auf der Strecke ab.

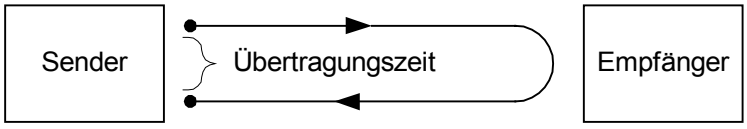


Abbildung 4.5: Übertragungszeit

Davon ist die maximale Bearbeitungszeit (worst case) die entscheidende Größe, ob die jeweilige Anforderung nach einem Echtzeitbetrieb erfüllt werden kann. Diese Zeit hängt nicht nur von internen Kommunikationsmechanismen sondern auch von verschiedenen Außenfaktoren ab, z.B. Anzahl der zu übertragenden Signale. Die Transportzeit auf der Strecke ist hier immer konstant. Um sowohl Sender als auch Empfänger zu berücksichtigen, werden in dieser Arbeit immer der arithmetische Mittelwert der Übertragungszeit, die aus mehreren Messungen mit unterschiedlichen Datenpaketen ermittelt wird, genommen:

$$\overline{T}(n_1,n_2)=\frac{1}{n_2-n_1+1}\sum_{k=n_1}^{n_2}T(k)\,.$$

Varianz

Die Varianz der Übertragungszeit gibt Auskunft über die mittlere Abweichung der Einzelwerte vom Mittelwert /Fil94/.

$$\sigma^2(n_1,n_2)=\frac{1}{n_2-n_1+1}\sum_{k=n_1}^{n_2}\left[T(k)-\overline{T}(n_1,n_2)\right]^2$$

Es ist wünschenswert, daß die Daten in einem regelmäßigen Zeitabstand bei Empfänger-Automatisierungssystem ankommen. Bei unregelmäßiger Übertragungszeit wird die Systemdynamik gestört.

Fehlerquote

Fehlerquote ist eine prozentuale Kenngröße, die ein Maß dafür ist, wieviele Datenpakete bei der Übertragung verlorengegangen sind (Abbildung 4.6).

$$E(n_1,n_2)=\frac{1}{n_2-n_1+1}\sum_{k=n_1}^{n_2}F(k)\qquad\begin{cases}F(k)=0, & falls\ \ u(k)_{send}=u(k)_{empfang}\\ F(k)=1, & falls\ \ u(k)_{send}\neq u(k)_{empfang}\end{cases}$$

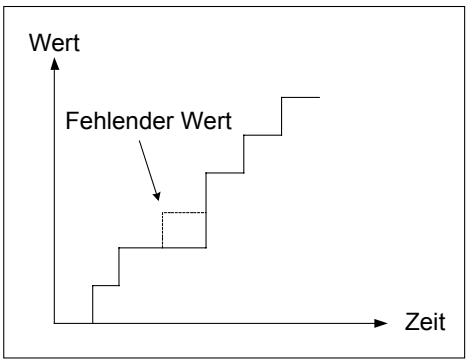


Abbildung 4.6: Übertragungsfehler

Einflußgrößen

Frequenz der Wertänderung

Der zu übertragende Prozeßwert ändert sich in einer unbestimmten Frequenz. Falls die Frequenz der Wertänderung sich erhöht, d.h. der Prozeßwert ändert sich schnell, beansprucht der Kommunikationsmechanismus auch mehr Rechenleistung, weil mehr Prozeßwerte übertragen werden müssen. Es ist wichtig zu wissen, wie die Frequenz der

Wertänderung sich auf die Leistungskenngrößen auswirkt. Man ermittelt damit auch die Grenze, bis zu der eine akzeptable Datenübertragung garantiert werden kann.

In dieser Arbeit wird der zu übertragende Prozesswert in bestimmter Takt geändert. Durch Manipulierung der Taktzeit wird die Frequenz der Wertänderung simuliert. Hier werden die ausgewählten Leistungskenngrößen gemessen, während die Taktzeit ΔT sich von 20 ms in Abstand von 20ms bis 200ms vergrößert. Als Kommunikationsfunktion wird BSEND/BRCV benutzt, die in OB1 mit niedrigster Priorität aufgerufen wird.

Als Eingangsgröße (zu übertragender Prozesswert) wird hier eine periodisch vorkommende Signalfolge eingesetzt. Beginnend mit 0 wird das Signal V je nach ΔT um ΔV (z.B. 1) vergrößert. Nach Tmax (z.B. 12 s) wird eine neue Folge gestartet (Abbildung 4.7).

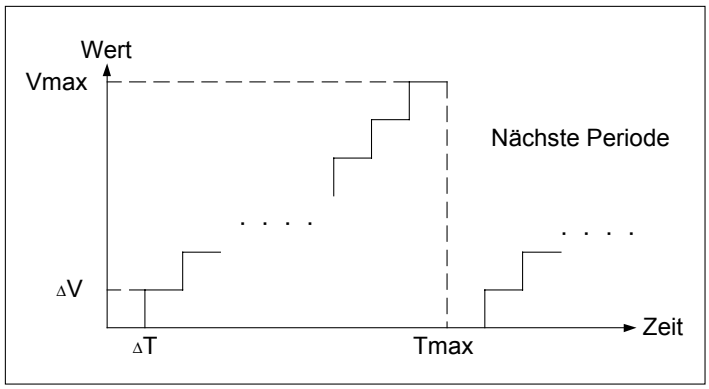


Abbildung 4.7: Eingangsgröße

Last

In einem Automatisierungssystem gibt es nicht nur Kommunikationsaufgaben, es gibt auch andere Anwendungen. Die Rechenleistung der CPU wird nach einer Prioritätsliste vergeben. Es ist sinnvoll zu wissen, welchen Einfluß auf die Leistungskenngrößen haben wird, wenn ein Kommunikationsmechanismus durch andere höherpriore Anwendungen verzögert wird. Last in dieser Arbeit bedeutet eine durch andere Anwendungen entstehende zeitliche Verzögerung, bevor die Kommunikationsaufgabe ausgeführt werden kann.

Der Lastgeber ist im OB35 eingebaut und wird "Lastaufträge" regelmäßig an die CPU abschicken. Durch Eingangsparameter kann man die Last zwischen 1 und 90 einstellen, wobei die größere Zahl sich auf eine größere Last bezieht. In der Abbildung 4.8 wird diese Abhängigkeit verdeutlicht. Je nach dem Lastauftrag kann T einen entsprechenden Wert (z.B. 1 ms) annehmen.

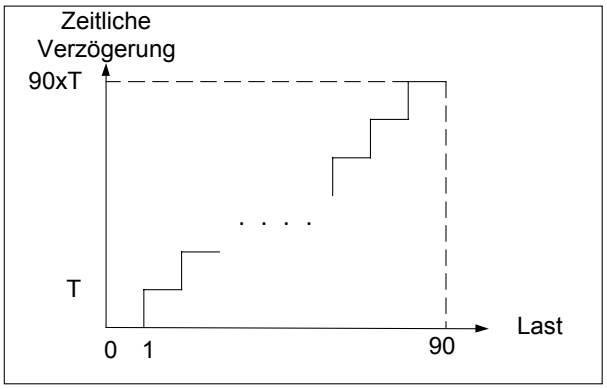


Abbildung 4.8: Lastgenerator

OB35 (Weckalarm - 100 ms Takt) für Lastgeber hat eine höhere Priorität als OB1 (Freier Zyklus) für Kommunikationsaufgabe. Die Folge ist, daß OB1 alle 100 ms von OB35 unterbrochen wird. Wie lange die Unterbrechung dauert, hängt von dem Stellwert des Lastgebers (1 bis 90) ab. Mit größer werdender Last wird immer weniger werdender Rechenzeit für OB1 bereitstehen (Abbildung 4.9). Hier wird der Signalgeber in OB34 eingebaut, wo eine Taktzeit von 200 ms eingestellt wird.

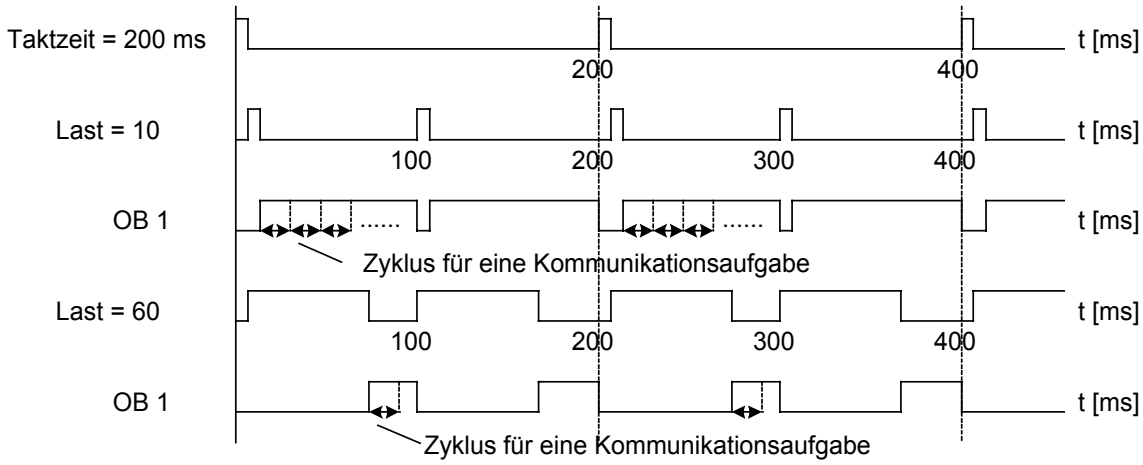


Abbildung 4.9: Einfluß der Last auf die Kommunikation

Die Funktionsweise des Versuchs ist in der Abbildung 4.10 dargestellt.

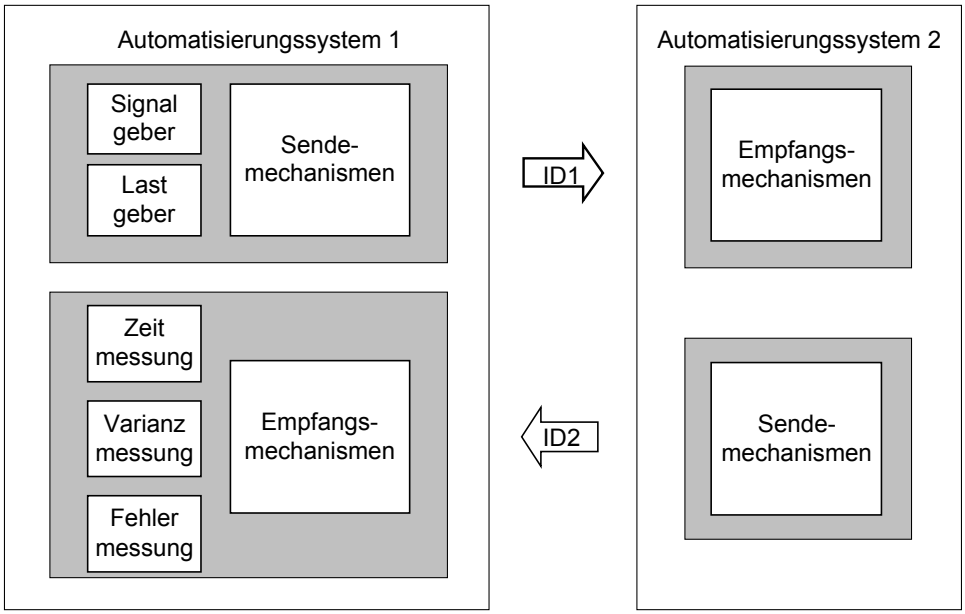


Abbildung 4.10: Blockbild des Versuchs

5 Explizite Verwendung spezifischer Kommunikationsbausteine

Die erste untersuchte Methode, die Kommunikation zwischen AS zu projektieren, besteht in der **expliziten Verwendung von spezifischen Kommunikationsbausteinen**. Mit „Explizit“ ist gemeint, daß der Anwender alle Kommunikationsbausteine manuell projektieren muß.

Für die Kommunikation zwischen SIMATIC S7/M7-400-Stationen werden die in FUP, KOP, AWL oder SCL programmierten Kommunikationsbausteine angeboten. Bei der programmgesteuerten Kommunikation kann man die gewünschte Funktionalität, d.h. den Zeitpunkt, die Datenmenge und das Übertragungsverfahren, durch Aufruf eines Kommunikationsbausteins im Anwenderprogramm explizit bestimmen. Da diese Anwendungsbausteine einem Dienst der Anwendungsschicht (Schicht 7 des ISO-Referenzmodells) entsprechen, sind sie unabhängig vom benutzten Medium und können auf allen Subnetzen (MPI, PROFIBUS, Industrial Ethernet) verwendet werden. In den folgenden Unterkapiteln werden die wichtigsten Kommunikationsbausteine nach der Funktionalität in unterschiedliche Kategorien eingeteilt. Erklärt wird auch die Funktionsweise der Bausteine, die zum Austausch der Nutzdaten eingesetzt werden. Abschließend wird die Performance dieser Methode gemessen und ausgewertet.

5.1 Kommunikationsbausteine

Nach ihrem Aufgabenbereich sind Kommunikationsbausteine in 3 Klassen einteilbar:

- Kommunikationsfunktionen,
- Programmmanagementfunktionen,
- Serverfunktionen.

Die Kommunikations- und Programmmanagementfunktionen sind weitestgehend genormt (Draft IEC/ISO 9506 Manufacturing Message Specification, Part 5: Companion Standard for Programmable Controller und Draft IEC 65A Programmable Controller, Part 3: Programming Languages). Damit wird erreicht, daß diese Funktionen in einem breiten Gerätespektrum (auch zur Kommunikation mit Fremdsystemen) eingesetzt werden können.

5.1.1 Kommunikationsfunktion

Die Kommunikationsfunktion dient zum Datenaustausch zwischen den Kommunikationspartnern. Von der Struktur her kann sie in zwei Kategorien eingeteilt werden: zweiseitige Kommunikationsfunktion und einseitige Kommunikationsfunktion.

Zweiseitige Kommunikationsfunktion

Eine Kommunikationsfunktion wird zweiseitig genannt, wenn der Datenaustausch im Anwenderprogramm der beiden Kommunikationspartner ausprogrammiert wird (Abbildung 5.1). Dieses Vorgehen erlaubt eine mit dem Anwenderprogramm koordinierte Übernahme der Daten auf der Sendeseite und auf der Empfangsseite. Zu dieser Kategorie gehören z.B. BSEND/BRCV, USEND/URCV, die sich in der Übertragungssicherheit und übertragbarer Datenmenge unterscheiden. Deren Funktionalität wird in Kapitel 5.3 genauer beschrieben.

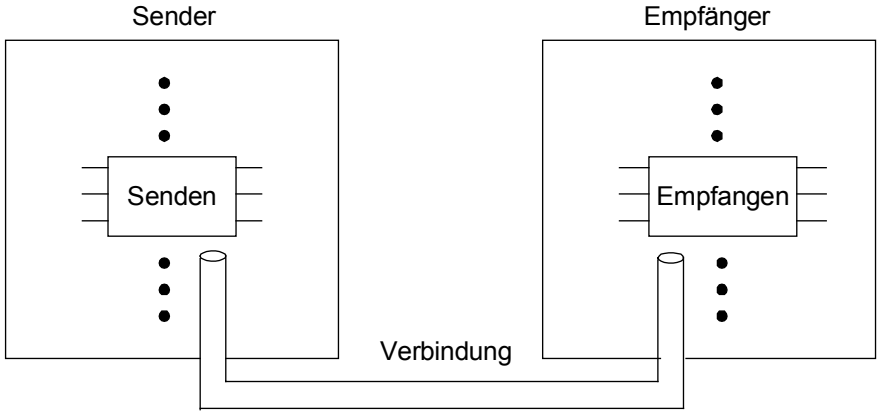


Abbildung 5.1: Zweiseitige Kommunikationsfunktion

Einseitige Kommunikationsfunktion

Es wird von einer einseitigen Kommunikationsfunktion gesprochen, wenn die Ausführung durch einen Funktionsaufruf vom Client programmiert wird, während im Anwenderprogramm des Servers für diese Funktion keine Programmierung nötig ist (Abbildung 5.2). Das Betriebssystem erbringt die Funktionalität auf der Serverseite asynchron zum Anwenderprogramm. GET, PUT, X_GET und X_PUT sind einseitige Funktionen zum Lesen und Schreiben von S7-Variablen beim Kommunikationspartner.

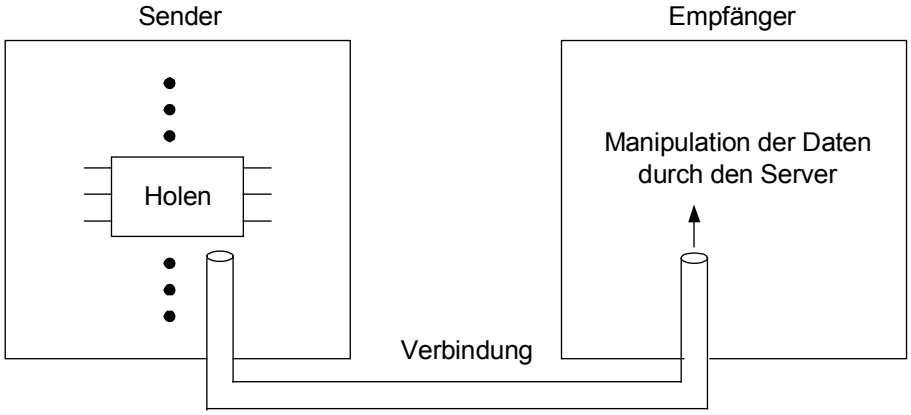


Abbildung 5.2: Einseitige Kommunikationsfunktion

5.1.2 Programmmanagementfunktion

Mit der Programmmanagementfunktion werden ausgewählte Zustände eines entfernten Gerätes überwacht und gegebenenfalls gesteuert. Dadurch gehört diese Funktion zu der Kategorie der einseitigen Funktion, da die Funktionalität bei dem entfernten Automatisierungssystem wiederum durch den Server erbracht wird.

Steuerungsfunktionen sind z.B. START, STOP, mit denen entsprechende Schalterbedienungen (an der Frontplatte) auf eine entfernte CPU per Kommunikation übertragen werden. Für das Abfragen des Zustands einer entfernten CPU steht die Steuerungsfunktion STATUS zur Verfügung.

5.1.3 Datenserverfunktion

Mit Hilfe von Datenserverfunktion können Datenfiles oder einzelne Datensätze in Datenfiles bearbeitet werden. Die Projektierungsarbeit hier richtet sich nur an die Anforderungsseite (Abbildung 5.3). Die Steuerungsfunktion OPEN/CLOSE steht für die Öffnung oder Schließung einer Datei. Die Steuerungsfunktion READS/WRITN wird beim Lesen oder Schreiben eines Datensatzes eingesetzt.

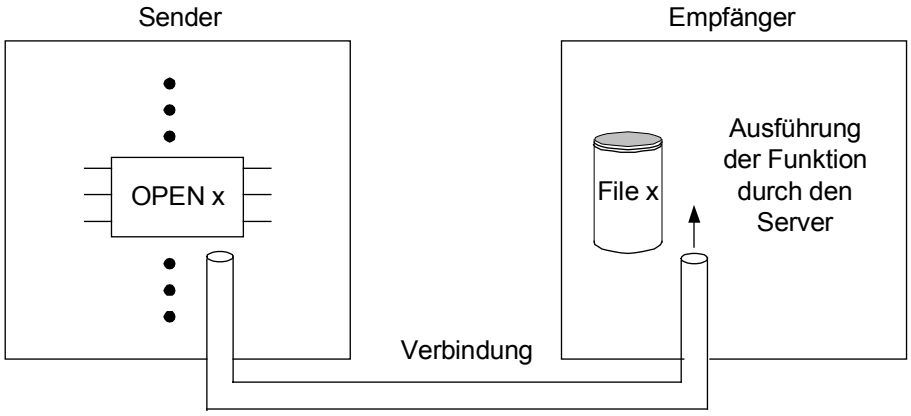


Abbildung 5.3: Datenserverfunktion

5.2 Parameter eines Kommunikationsbausteins

Um eine konfliktfreie Datenübertragung zu gewährleisten, braucht jeder Baustein einige zusätzliche Informationen, welche vom Anwender projektiert werden müssen. Diese Informationen sind die Parameter der Kommunikation. Sie sind in der Abbildung 5.4 zusammengefaßt dargestellt.

Adressierungsparameter dienen der Adressierung des entfernten Kommunikationspartners. Eine eindeutige Kommunikationsbeziehung zwischen den Automatisierungssystemen ist zwingend notwendig.

Steuerungsparameter werden zur Aktivierung der Funktionsausführung eines Bausteins genutzt. Ein Kommunikationsbaustein wird in der Regel nicht implizit durch den Bausteinaufruf aktiviert. Vielmehr muß die Aktivierung der einzelnen Funktionen eines Bausteins durch Steuerungsparameter explizit erfolgen.

Zustandsparameter dienen der Überwachung des Bausteinzustandes. Sie deuten auch an, ob der Auftrag fehlerfrei bearbeitet wird.

Sendeparameter referenziert diejenigen Daten aus der S7-Datenwelt, welche zum entfernten Partner gesendet werden sollen.

Empfangsparameter referenziert die Datenbereiche der S7-Datenwelt, in welcher die vom entfernten Partner empfangenen Daten eingetragen werden.

Abbildung 5.4: Parameter eines Kommunikationsbaustein

5.3 Beschreibung der Kommunikationsfunktion

Zu den wichtigsten Kommunikationsfunktionen gehören BSEND/BRCV und USEND/URCV. Die jeweiligen Eigenschaften sowie Funktionsweisen werden im folgenden genauer beschrieben.

5.3.1 BSEND/BRCV

Die Steuerungsfunktion BSEND (Blockorientiertes Senden) sendet Daten über MPI, PROFIBUS oder eine Industrial Ethernet-Verbindung zu einer weiteren S7 CPU, welche den Funktionsbaustein BRCV (Blockorientiertes Empfangen) zum Empfang der Daten aufrufen muß. Als Datenquelle bzw. Datenziel sind dabei nur Datenbausteine erlaubt. In STEP7 ist dafür beidseitig eine homogene Transportverbindung zu projektieren und in das Automatisierungsgerät zu laden.

Die Steuerungsfunktion BSEND sollte verwendet werden, wenn der Datenumfang größer als 440 Byte ist, oder wenn bei einem Datenumfang von weniger als 440 Byte eine gesicherte Übertragung gewünscht wird. Eine gesicherte Übertragung heißt, die Datenübertragung ist erst dann abgeschlossen, wenn die Empfangsfunktion im Kommunikationspartner die Daten übernommen hat. Durch die automatische Wiederholung von möglichen unvollständigen oder falschen Telegrammen auf dem MPI/PROFIBUS oder dem Industrial Ethernet (Schicht 2 des ISO-Referenzmodells) wird eine hohe Datensicherheit erreicht.

Da die Datenübertragung segmentiert erfolgt, liegen die Daten nur nach Abschluß des Auftrags (d.h. nach Eintreffen der Quittung) konsistent im Empfangs-DB vor.

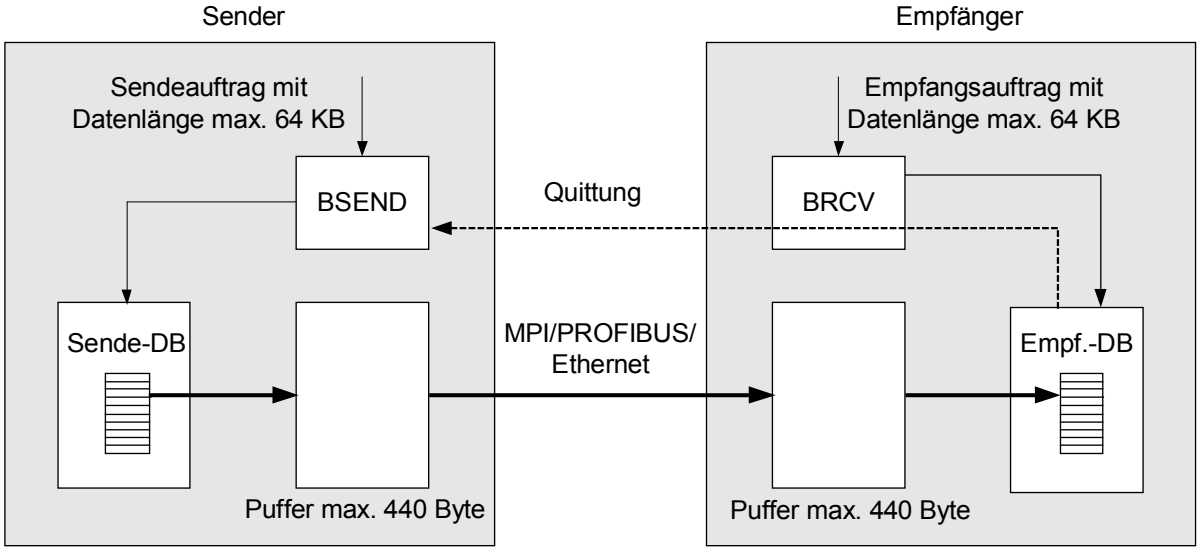


Abbildung 5.5: Kommunikationsablauf bei BSEND/BRCV

Mit dem verwendeten BSEND/BRCV kann eine größere Datenmenge (z.B. gesamter oder größerer Teilbereich eines Datenbausteins, Merkerbereichs oder Prozeßabbildes) - bis 64 kByte - zwischen SIMATIC S7/M7-400-Stationen transferiert werden als dies mit allen anderen Kommunikations-SFBs für projektierte Verbindungen möglich ist. Das liegt daran, daß der zu sendende Datenbereich in Einheiten von 440 Byte segmentiert wird. Jedes Segment wird vom Betriebssystem der CPU einzeln an den Partner gesendet und dort automatisch vom Betriebssystem in den Empfangs-DB eingetragen. Vor Absenden des nächsten Segmentes wird die betriebssysteminterne Quittung des gerade gesendeten Segments abgewartet (Abbildung 5.5).

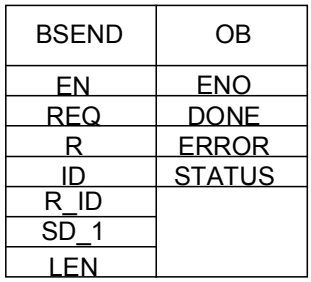


Abbildung 5.6: Kommunikationsfunktion BSEND

Die Aktivierung des Sendevorgangs erfolgt durch einen Aufruf des Bausteins mit einer positiven Flanke von 0 zu 1 am Steuereingang REQ (Abbildung 5.6). Deswegen muß BSEND zweimal aufgerufen werden, um erneut senden zu können (BSEND benötigt zur Aktivierung einen Flankenwechsel von 0 auf 1 am Steuereingang REQ). Sinnvoll ist diese Sendeauftragsvergabe jedoch nur dann, wenn das Telegramm zwischen zwei Funktionsaufrufen zeitlich übertragen werden kann, genauer gesagt, wenn der Aufruf des BRCV in dem Empfänger-Automatisierungssystem schneller ist als der des BSEND in dem Sender-Automatisierungssystem (Das Empfänger-Automatisierungssystem benötigt für jedes Telegramm zwei Aufrufe des BRCV).

Die Anfangsadresse der zu sendenden Daten wird durch SD_1 vorgegeben, die Länge des Datenblocks durch LEN (hier bis zu 64 K). Das Lesen der Daten aus dem Anwenderspeicher erfolgt asynchron zur Bearbeitung des Anwenderprogramms. Die kleinste zusammenhängende Datenmenge die vom Betriebssystem konsistent aus dem Anwenderspeicher gelesen wird, beträgt 32 Byte bei den CPUs 4xx. Soll auch an den 32-Byte Grenzen Konsistenz erreicht werden, so dürfen nach dem Aufruf des BSEND die Daten im Sende-DB nicht verändert werden, solange der Auftrag läuft. Der erfolgreiche Abschluß des Sendevorgangs wird am Zustandsparameter DONE mit 1 angezeigt.

Aufgrund der asynchronen Datenübertragung kann ein erneutes Senden von Daten erst gestartet werden, wenn die vorhergehenden Daten durch Aufruf des Partner–SFB abgeholt wurden. Bis die Daten abgeholt wurden, wird beim Aufruf des BSEND der Statuswert 7 ausgegeben. Bei einer positiven Flanke am Steuereingang R wird ein laufender Sendevorgang abgebrochen.

Bei fehlerhafter Übertragung zeigen die Ausgänge ERROR und STATUS spezifische Fehlerinformationen an, welche dem BSEND entsprechen.

Die Parameter ID und R_ID legen eindeutig die Kommunikationsbeziehung zwischen den Automatisierungssystemen fest.

BRCV	OB
EN	ENO
EN_R	NDR
ID	ERROR
R_ID	STATUS
RD_1	
LEN	

Abbildung 5.7: Kommunikationsfunktion BRCV

Das Eintragen der Daten in den Anwenderspeicher erfolgt auch asynchron zur Bearbeitung des Anwenderprogramms. Die Anfangsadresse des Zielbereichs der empfangenen Daten wird durch RD_1 vorgegeben (Abbildung 5.7). Der Parameter LEN gibt bei jedem Zyklus die Anzahl der bereits empfangenen Daten in Bytes an.

Die kleinste zusammenhängende Datenmenge, die vom Betriebssystem konsistent in den Anwenderspeicher geschrieben wird, beträgt 32 Byte. Soll auch an den 32-Byte Grenzen Konsistenz erreicht werden, so dürfen nach Aufruf des BRCV die Daten im Empfangs-DB solange nicht bearbeitet werden, solange der Auftrag läuft (NDR = 0). Ist der Auftrag ohne Fehler beendet, wird der Ausgang NDR für einen Zyklus auf 1 gesetzt. Im Folgezyklus wird automatisch wieder die Empfangsfreigabe an das Betriebssystem der CPU gegeben (ab und inklusive diesem Aufruf ist NDR wieder 0). Die Empfangsfreigabe kann bereits wirksam werden, bevor der erste Empfangsauftrag eintrifft, dann wird sie vom Betriebssystem gespeichert. Für jedes Telegramm werden somit mindestens zwei Aufrufe des BRCV benötigt (Abbildung 5.8).

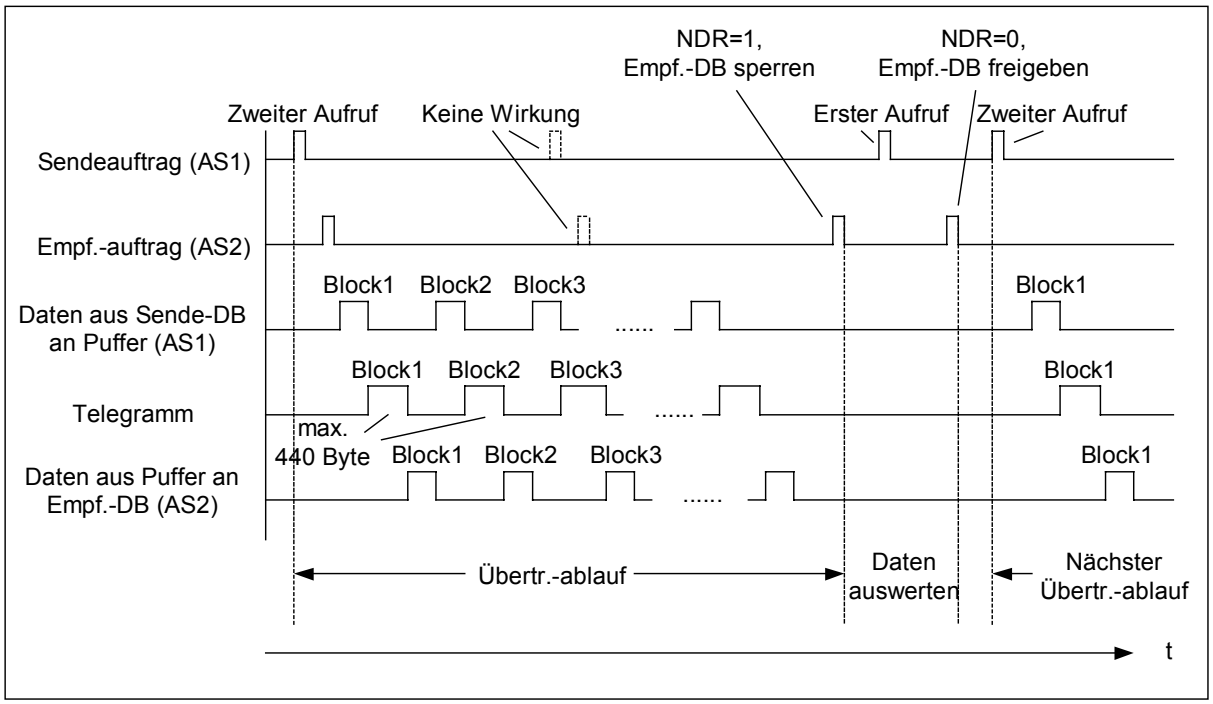


Abbildung 5.8: Ablauffolge der Übertragung (BSEND/BRCV)

Die anderen Anschlüsse wie ID, R_ID haben die gleiche Bedeutung wie bei BSEND.

5.3.2 USEND/URCV

Wie BSEND sendet auch USEND (Unkoordiniertes Senden) Daten über MPI, PROFIBUS oder Industrial Ethernet-Verbindung zu einer weiteren S7 CPU, (mit MPI auch M7 oder M7-S7), welche den Funktionsbausteintyp URCV (Unkoordiniertes Empfangen) zum Empfang der Daten aufrufen muß.

USEND kann statt BSEND verwendet werden, wenn der Datenumfang kleiner gleich 440 Byte ist, Datenkonsistenz über den ganzen Sendedatenbereich ständig verlangt wird und keine gesicherte Übertragung gefordert ist.

Im Empfangs-DB liegen die Daten ständig konsistent vor (im Gegensatz zum BSEND). Dies wird dadurch erreicht, daß die Daten bereits beim Aufruf der Bausteine USEND und URCV von diesen vollständig und direkt in den Puffer - bzw. aus dem Puffer - des Betriebssystems kopiert werden. Inkonsistenz kann nur dann auftreten, wenn der Kopiervorgang durch ein Alarmprogramm unterbrochen wird (entspricht Unterbrechung des unterlagerten USEND bzw. URCV) und im Alarmprogramm die Sende-/Empfangsdaten verändert werden.

Die Übertragung von Daten ist unabhängig von der Bearbeitung der Kommunikationsfunktion (URCV) beim Kommunikationspartner. Beim USEND/URCV wird die Quittung auf Betriebssystemebene ohne laufende Koordinierung mit dem Anwenderprogramm erzeugt. D.h. die Quittung auf den USEND trifft auch dann ein, wenn die Daten nicht in den Empfangs-DB eingetragen werden konnten, z.B. wenn die Empfangs-CPU in STOP ist. Die Folge ist dann, die Daten können beim Kommunikationspartner durch aktuellere Daten überschrieben werden (Abbildung 5.9).

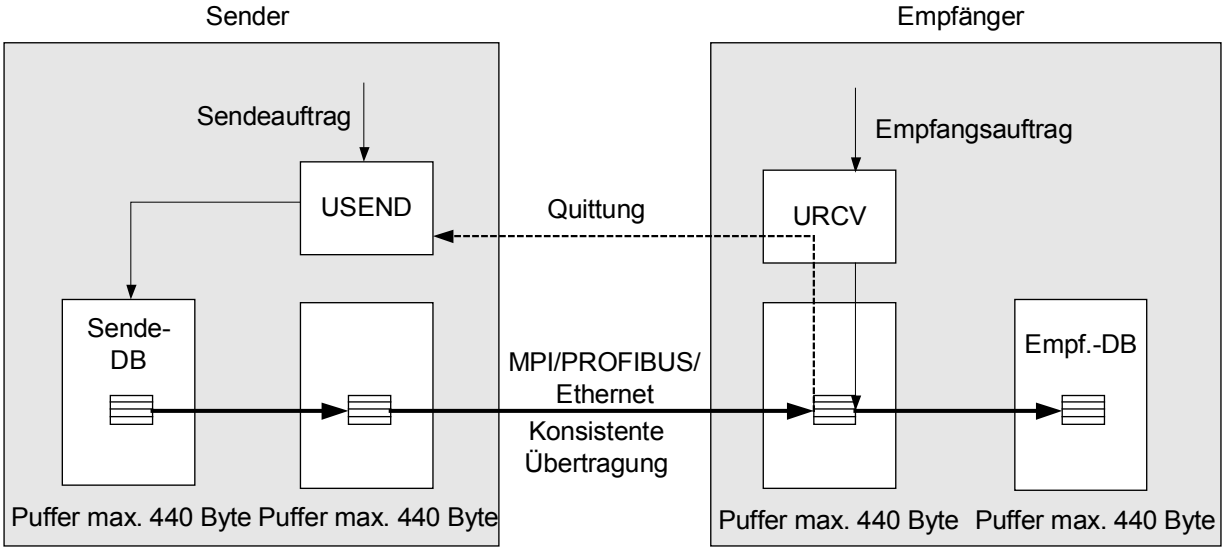


Abbildung 5.9: Kommunikationsablauf bei USEND/URCV

Arbeitsweise des Bausteins

Mit dem intern verwendeten USEND (Abbildung 5.10) kann eine Datenmenge von 440 Byte konsistent zwischen den Kommunikationspartnern übertragen werden. Der Sende-Vorgang verläuft bzgl. der Quittung ohne Koordination mit dem Empfangs-Funktionsbaustein, d.h. die Quittung wird vom Betriebssystem des Empfängers generiert, ohne daß auf den Aufruf des Empfangsbausteins gewartet wird.

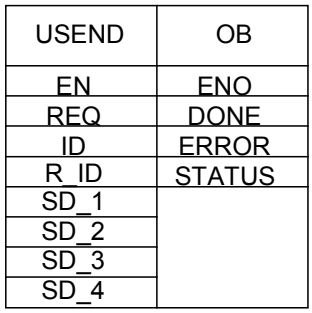


Abbildung 5.10: Kommunikationsfunktion USEND

USEND sendet Daten an URCV auf einem entfernten Partner (der Parameter R_ID muß bei beiden SFBs identisch sein). Der Sendevorgang erfolgt nach einer positiven Flanke am Steuereingang REQ. Er verläuft ohne Koordination mit dem Partner-SFB. Die zu sendenden Daten werden durch die Parameter von SD_1 bis SD_4 referenziert, wobei diese vier Sendeparameter nicht alle belegt sein müssen. Der Anwender muß jedoch darauf achten, daß die über die Parameter SD_i und RD_i, $1 < i < 4$, definierten Bereiche in der Anzahl, in der Länge und im Datentyp zueinander passen (RD_i gehört zum zugehörigen Partner-SFB "URCV" (Abbildung 5.11).). Der erfolgreiche Abschluß des Sendevorgangs wird am Zustandsparameter DONE mit 1 angezeigt.

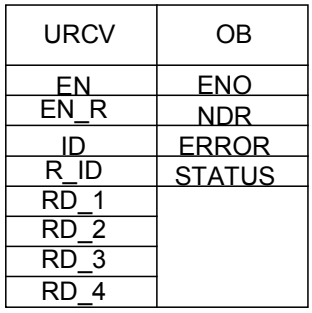


Abbildung 5.11: Kommunikationsfunktion URCV

Falls beim Aufruf am Steuereingang EN_R des URCV 1 anliegt, werden die empfangenen Daten in die projektierten Empfangsbereiche kopiert. Diese Datenbereiche werden durch die Parameter von RD_1 bis RD_4 referenziert. Beim Erstaufruf wird das "Empfangsfach" angelegt. Bei allen weiteren Aufrufen müssen die zu empfangenden Daten in dieses Empfangsfach hineinpassen. Der Abschluß des Kopiervorgangs wird am Zustandsparameter NDR mit 1 angezeigt (Abbildung 5.12).

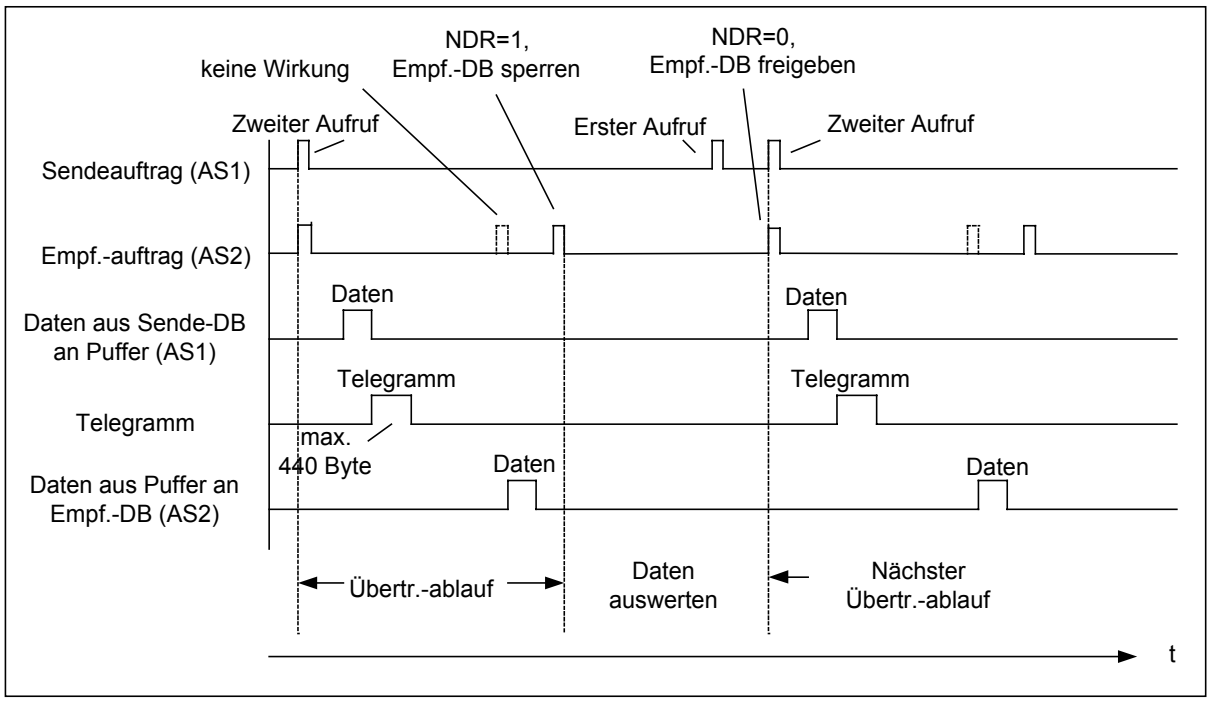


Abbildung 5.12: Ablauffolge der Übertragung (USEND/URCV)

5.4 Usability

Die Projektierung der Verbindung läuft in diesem Verfahren explizit. In einem Projektierungsplan sind für die Kommunikation 2 Schritte für jede Verbindung vorzunehmen:

- Einfügen der Kommunikationsbausteine für alle zu übertragende Signale,
- Einstellung der Parameter für alle eingesetzten Kommunikationsbausteine.

5.4.1 Einfügen des Kommunikationsbausteins

Die Projektierungsmethode sieht vor, für jede Verbindung zwischen den CPUs entsprechende Kommunikationsbausteine in den Plan einzufügen. Die Bausteine bieten eine Schnittstelle in der Anwendungsschicht für den Anwender. Der Kommunikationsdienst in den unteren Schichten wird dadurch „unsichtbar“.

Bei diesem Konzept werden die Kommunikationsdienste ausschließlich durch vorprogrammierte Bausteine angeboten. Als Beispiel wird hier ein einfacher Regelkreis betrachtet. Wie in der Abbildung 5.13 gezeigt soll die Regelkette auf 2 CPUs verteilt werden. CPU1 übernimmt die Funktion des Reglers, CPU2 simuliert die Strecke. Stellgröße und Regelgröße sollen zwischen den beiden CPUs ausgetauscht werden.

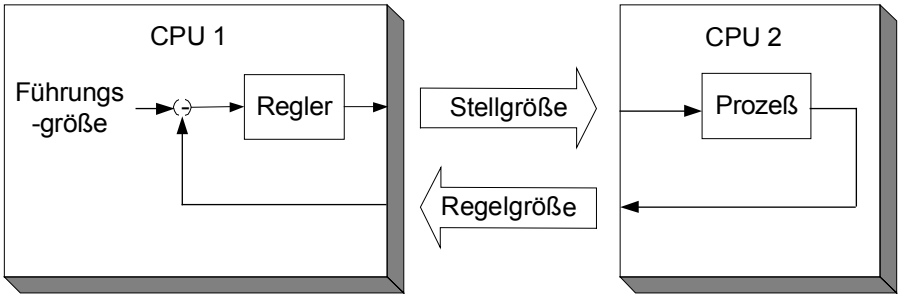


Abbildung 5.13: Verteilung eines Regelkreises auf zwei CPUs

Um Regelgröße und Stellgröße von dem jeweils anderen Kommunikationsteilnehmer zu bekommen, ist die Projektierung der geräteübergreifenden Kommunikation notwendig. Hier werden jeweils die Kommunikationsbausteine im Plan plaziert.

Die Festlegung einer Kommunikationsbeziehung zwischen zwei Bausteinen auf unterschiedlichen CPUs wird dadurch erreicht, daß

- für jede Kommunikationsbeziehung Kommunikationsfunktionen, jeweils für Senden und für Empfangen, explizit in den Plänen auf Sender und Empfänger platziert werden.
- eine eindeutige Zuordnung der Kommunikationsbeziehung zwischen CPUs durch Bausteinparameter V_ID und R_ID realisiert wird.

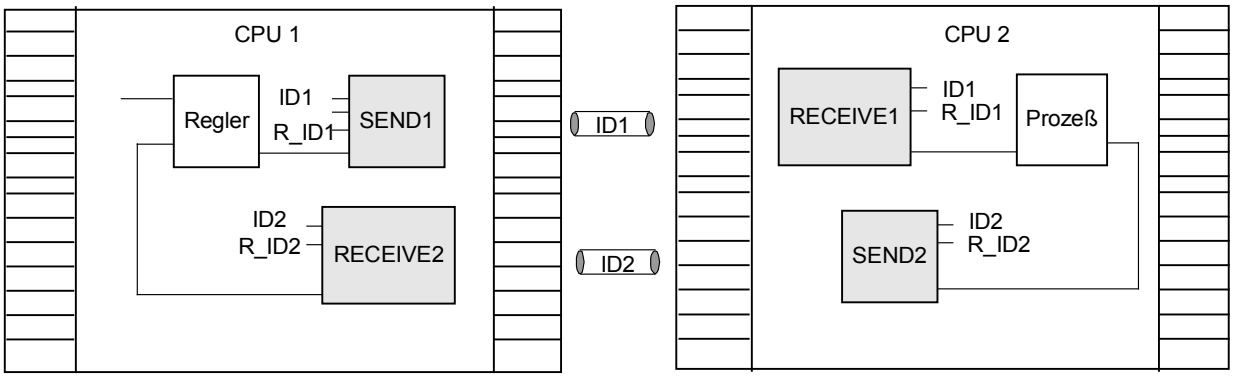


Abbildung 5.14: Kommunikationsprojektierung für einen Regelkreis

5.4.2 Einstellung der Kommunikationsparameter

Wie vorher erwähnt wird, gibt es an jedem Kommunikationsbaustein eine bestimmte Anzahl von Parametern, die von dem Anwender eingestellt werden müssen. Die wichtigsten davon sind die Adressenparameter ID und R_ID.

Parameter ID

Der Parameter ID referenziert eine logische Verbindung innerhalb eines Gerätes, z.B. einer CPU. Über die referenzierte Verbindung sendet oder empfängt der Kommunikationsbaustein die Daten. Zwischen zwei Automatisierungssystemen können mehrere Verbindungen mit zugehörigen Verbindungs-IDs hergestellt werden. Die ID ist von lokaler Ausprägung, d.h. eine Verbindung kann an ihren beiden Enden durch unterschiedliche IDs referenziert werden. Aber sie muß innerhalb eines Gerätes eindeutig vergeben werden.

Die Verbindungsbeschreibung selbst enthält alle notwendigen Informationen zur Adressierung des entfernten Gerätes. Die Verbindungsbeschreibung ist anwendungsspezifisch und wird mittels Projektierungswerkzeug (z.B. VerbPro in SIMATIC) bereitgestellt.

Eine zweiseitige Kommunikationsbeziehung ist dadurch gekennzeichnet, daß auf lokaler und entfernter Seite jeweils ein Baustein existiert. Die beiden Bausteine bilden zusammen ein Bausteinpärchen. Jeder der beiden Bausteine referenziert eine lokal vorhandene Verbindungsbeschreibung: Abbildung 5.15.

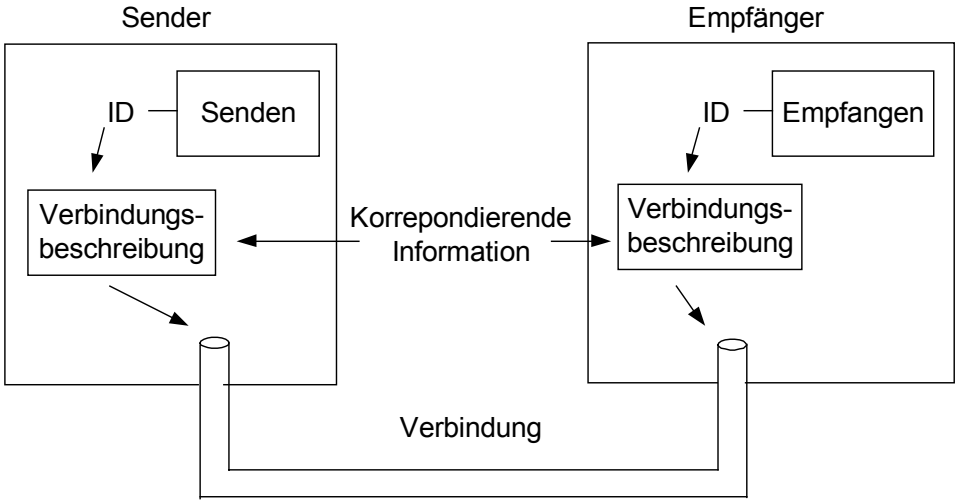


Abbildung 5.15: Zweiseitige Kommunikation

Bei einseitiger Kommunikation wird die Dienstleistung auf der entfernten Seite nicht durch einen Kommunikationsbaustein, sondern durch einen Server als Systemleistung erbracht. Eine Referenz durch die ID ist hier nicht gegeben. Die Verbindungsbeschreibung ist jedoch vorhanden: Abbildung 5.16.

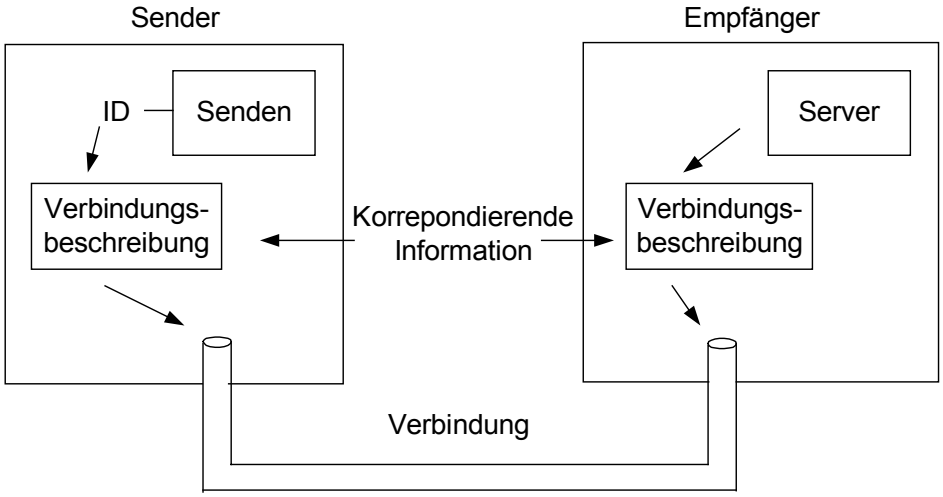


Abbildung 5.16: Einseitige Kommunikation

Parameter R_ID

Der Parameter R_ID referenziert bei zweiseitiger Kommunikationsbeziehung genau ein Bausteinpärchen, somit genau die funktionale Zusammengehörigkeit eines lokalen und eines entfernten Bausteins. In SIMATIC S7 sind die R_ID-Referenzen innerhalb einer logischen Verbindung eindeutig.

Über eine logische Verbindung können prinzipiell mehrere Bausteinbeziehungen existieren. Mit der R_ID wird die Zusammengehörigkeit eines Sende- und eines Empfangs-SFBs über dieselbe Verbindung festgelegt: Abbildung 5.17.

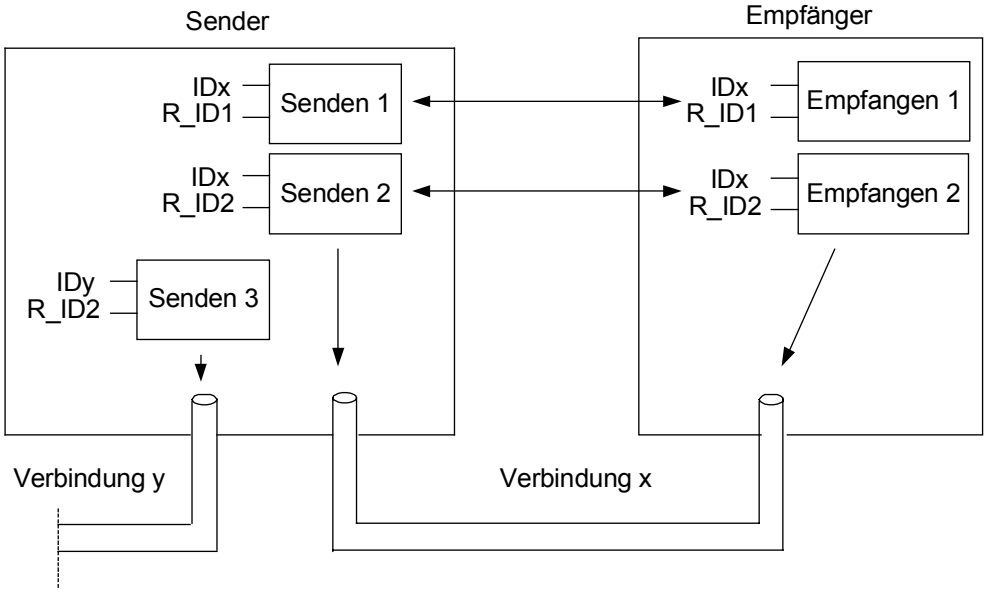


Abbildung 5.17: Referenzierung durch Adreßparameter R_ID

Die Anzahl der gleichzeitig abwickelbaren Bausteinbeziehungen ist prinzipiell beliebig. CPU-spezifische Restriktionen sind möglich.

5.4.3 Änderungsprojektierung

Einzelne Funktionspläne können zu jedem Zeitpunkt geändert werden. Die Änderung der Verschaltungen im Funktionsplan kann folgendes beinhalten:

- geräteübergreifende Verschaltung löschen,
- neue geräteübergreifende Verschaltung in den Funktionsplänen einrichten,
- geräteübergreifende Verschaltung umbelegen / umverdrahten.

Außerdem können geräteübergreifende Verbindungen zu lokalen und lokale zu übergreifenden Verbindungen werden. Hier müssen folgende Fälle betrachtet werden:

- lokale Verschaltung wird zur geräteübergreifenden Verschaltung;
- geräteübergreifende Verschaltung wird zur lokalen Verschaltung;
- aus der übergreifenden Verschaltung Automatisierungssystem x – Automatisierungssystem y wird die Verschaltung Automatisierungssystem x – Automatisierungssystem z.

Bei der Methode unter Verwendung von expliziten Kommunikationsbausteinen wird eine geräteübergreifende Kommunikationsverbindung damit gelöscht, daß die entsprechenden Kommunikationsbausteine gelöscht werden. Für die neue Verbindung sind die ganzen Projektierungsschritte durchzuführen. Nach der Übersetzung sind die Änderungen auf beide CPUs zu laden.

5.5 Meßergebnisse

Bevor die Kriterien in Kapitel 4.3 für die Methode unter Verwendung von expliziten Kommunikationsbausteinen ausgewertet werden, soll die benötigte Bearbeitungszeit für den Kommunikationsvorgang betrachtet werden. Diese Zeit gibt an, wie lange es maximal gezögert werden kann (worst case), um die Echtzeitbedingung (rechtzeitige Übertragung) nicht zu beeinflußen.

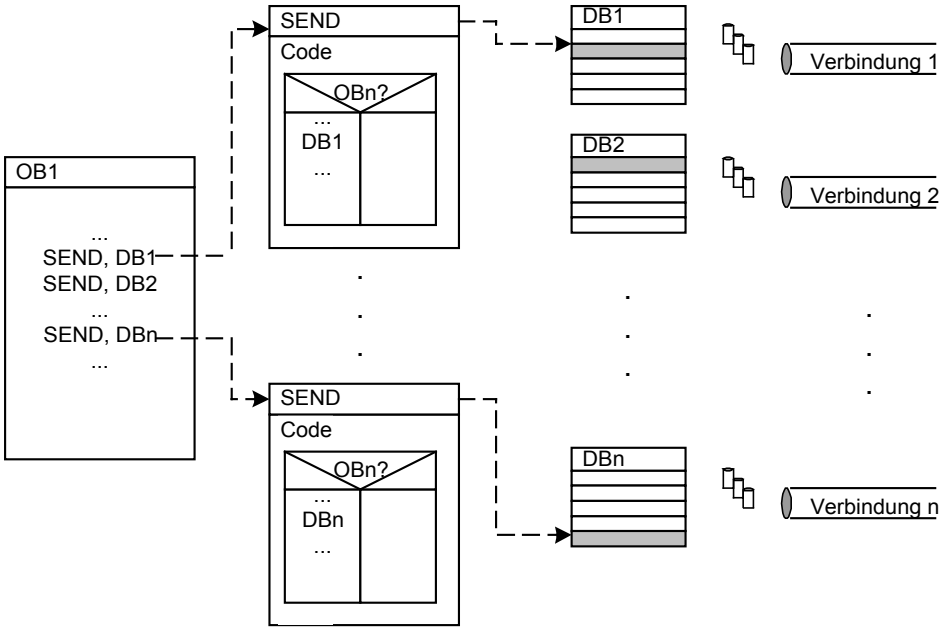


Abbildung 5.18: Bearbeitungsreihenfolge auf der Sendeseite (Explizite Methode)

In dieser Methode wird für ein Signal jeweils ein Sendevorgang angestoßen. Der Sendebaustein bearbeitet entsprechende Code-Abschnitte bei verschiedenen Ereignissen (unterschiedliche OB-Aufrufe), wobei die zu bearbeitenden Daten im jeweiligen DB liegen. Der Aufruf für den normalen Betrieb erfolgt in OB1 (Abbildung 5.18). Die Bearbeitungszeit hängt von der CPU-Leistung ab. Falls für die Übertragung eines Signal T_{SEND} benötigt wird, errechnet sich die gesamte Rechenzeit für n Signale auf der Sendeseite wie folgt,

$$t_{1S} = n \cdot T_{SEND}$$

Für die Empfangsseite ist entsprechend,

$$t_{1R} = n \cdot T_{RCV}$$

Falls der Kommunikationsvorgang nicht durch anderen Rechenvorgang mit höherer Priorität unterbrochen wird, muß t_{1S} oder t_{1R} gewartet werden, bis der Sende- oder Empfangsvorgang für das selbe Signal wieder angestoßen werden kann. Um die Empfangsseite nicht mit Sendedaten zu überfluten, soll auch $t_{1S} > t_{1R}$ garantiert werden.

5.5.1 Einfluß der Taktzeit des Signalgebers

Der Einfluß der Taktzeit des Signalgebers auf die Übertragungszeit und Varianz ist in der Abbildung 5.19 dargestellt.

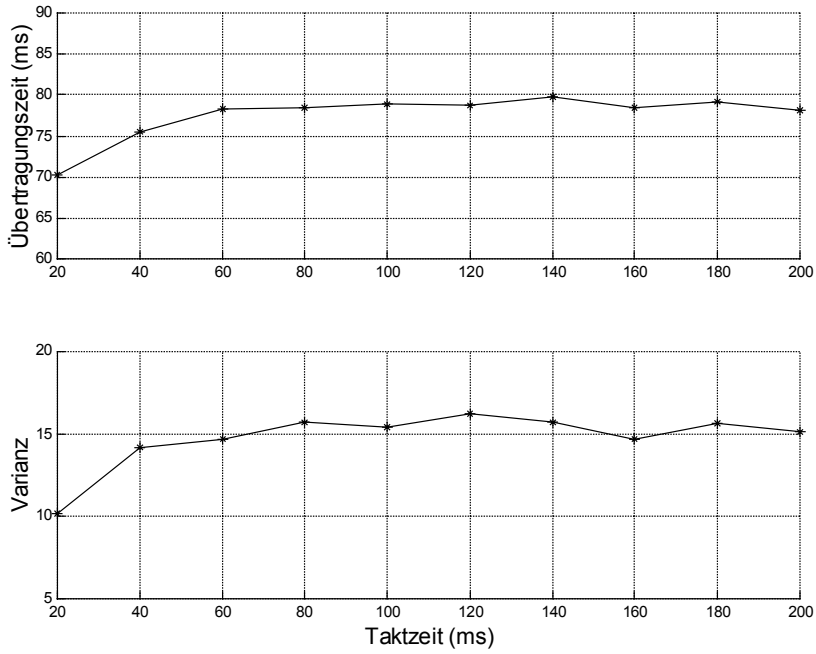


Abbildung 5.19: Übertragungszeit und Varianz in Abhängigkeit von der Taktzeit

Die Übertragungszeit bewegt sich um 80 ms. Varianz der Übertragungszeit ist um 15 ms^2. Die beiden Kenngrößen bleiben für unterschiedliche Taktzeiten fast konstant. Die Zeitschwankung kommt zum Teil dadurch, daß eine asynchrone Datenübertragung mit Rückmeldung (Handshake Verfahren) hier eingesetzt wird. Keine Synchronisierung besteht zwischen den Kommunikationspartnern. Für die Übertragung besteht somit kein starrer Zeitrahmen. Die unterschiedlichen Aufrufpunkte bestimmen auch eine unterschiedliche Übertragungszeit.

Bei kleineren Taktzeiten (z.B. 20 ms) hat der Prozeßwert sich so schnell geändert, daß nach der Änderung gleich ein Sendevorgang aufgerufen wird. Die Wartezeit bis zum Senden hat sich im Vergleich zu den größeren Taktzeiten verkleinert. Das kann auch erklären, wieso bei kleineren Taktzeiten (z.B. 20 ms) die Übertragungszeit auch kleiner wird. Die Varianz ist auch somit kleiner.

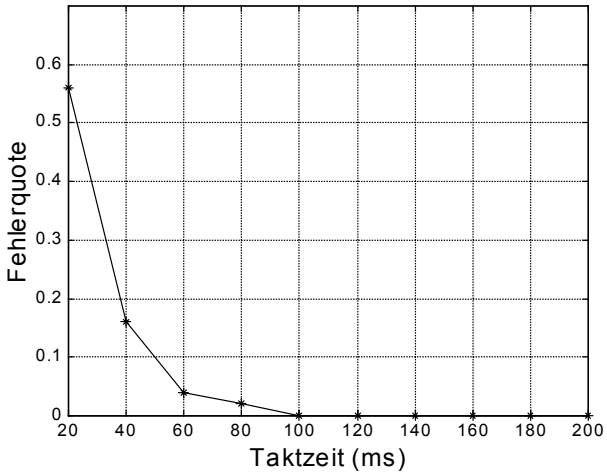


Abbildung 5.20: Fehlerquote in Abhängigkeit von der Taktzeit

Die Fehlerquote vergrößert sich mit immer kleiner werdender Taktzeit (Abbildung 5.20). Sie wird hauptsächlich durch folgende Punkte verursacht,

1. Nachdem der Sendebaustein einen Wert gesendet hat, muß er auf die Quittung warten. Die Quittung ist positiv im Fall von der Annahme des Telegramms, und negativ im Fall der Zurückweisung des Telegramms. Sie wird vom Empfänger-AS generiert. Das ist eine Systemleistung und bedarf keines externen Bausteinaufrufs. Vor der Entgegennahme der Quittung darf der Sendebaustein nicht senden. Falls der Prozeßwert sich schnell ändert (bei der kleineren Taktzeit) und gesendet wird, ergibt sich eine immer kürzere freie Rechenzeit für den Aufruf von Empfangsbaustein auf Empfänger-AS. Beim Ankommen eines Telegramms kann es durchaus sein, daß ein vorheriges Telegramm noch nicht von Empfänger-AS abgeholt wurde. So verweigert das Empfänger-AS, neue Daten anzunehmen. Als Antwort wird eine negative Quittung erzeugt (Abbildung 5.21).

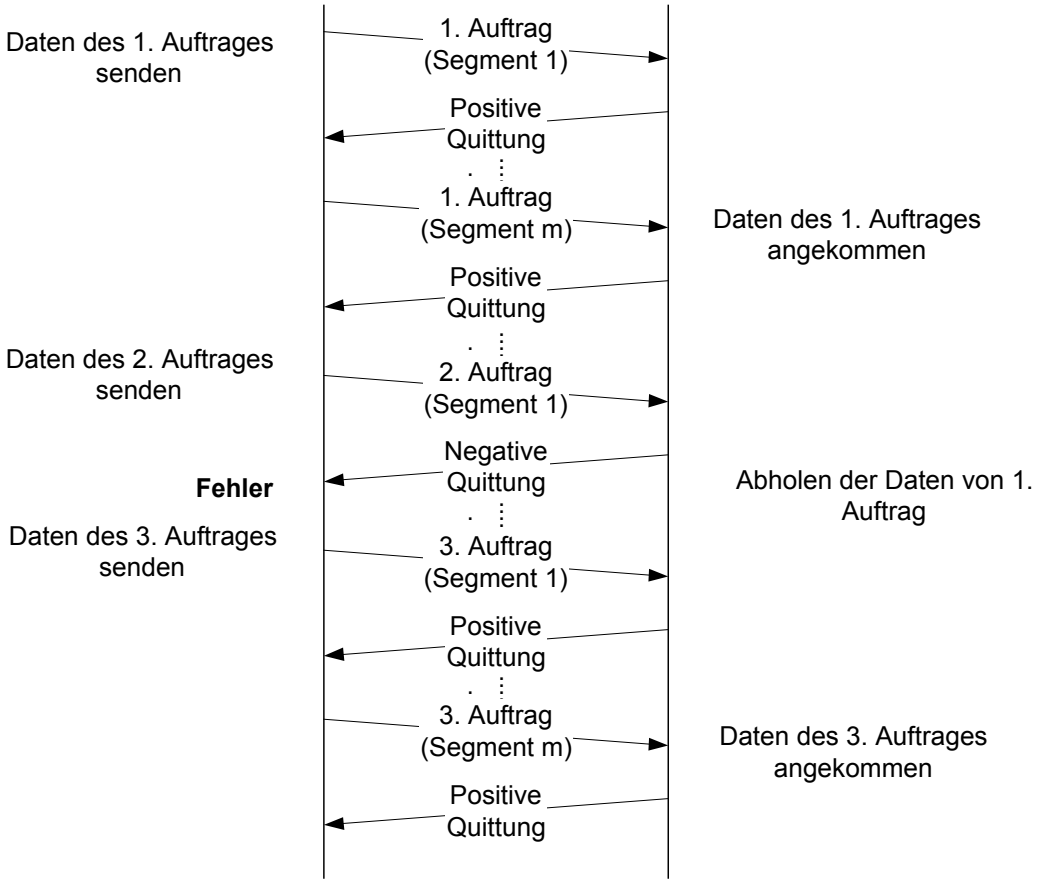


Abbildung 5.21: Fehler durch Zurückweisen durch Empfänger-Automatisierungssystem

Eine Möglichkeit, diese Art von Fehlern zu reduzieren, besteht darin, innerhalb einer Taktzeit des Signals den Kommunikationsbaustein möglichst häufig aufzurufen, damit das Sender-Automatisierungssystem mehrfach versuchen kann, den gleichen Inhalt zyklisch zu senden, bis das Empfänger-AS den Wert annimmt.

2. Eine andere Fehlerquelle ist: Die minimale Zeit zwischen zwei Übertragungsvorgängen beträgt ungefähr 50 ms. Das Signal kann sich nicht schneller ändern, als der Kommunikationsbaustein senden darf. Bei Unterschreiten dieser Schwelle wird der noch nicht gesendete Wert durch neue überschrieben. Bei einer Taktzeit um 50ms kommen deswegen mehr Fehler. Je kleiner der Takt ist, desto mehr wird der nicht rechtzeitig gesendete Wert durch neue überschrieben.

5.5.2 Einfluß der Last

Das Testergebnis für den Einfluß der Last auf die Übertragungszeit und die Varianz wird in der Abbildung 5.22 dargestellt,

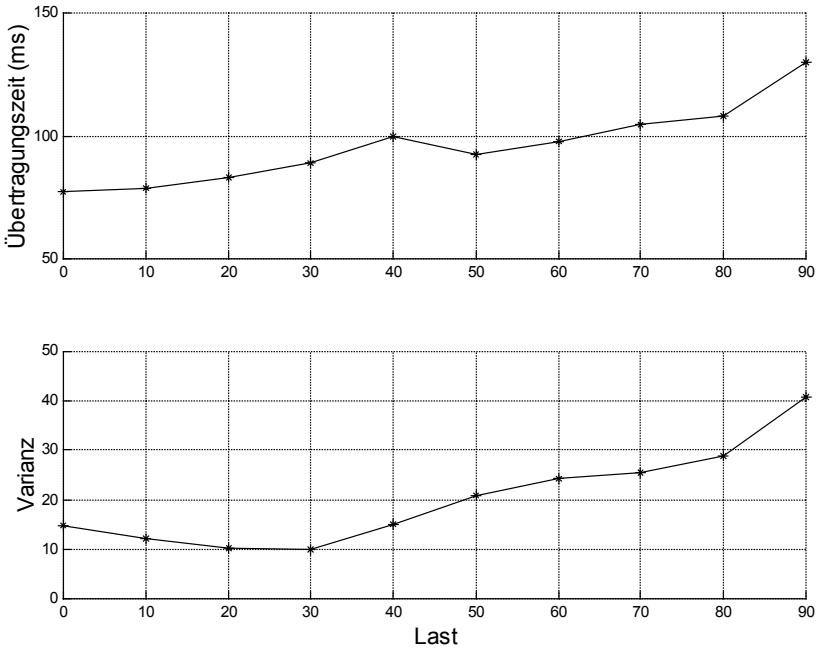


Abbildung 5.22: Übertragungszeit und Varianz in Abhängigkeit von der Last

Der Aufruf von Sendebaustein und Empfangsbaustein kann erst ausgeführt werden, nachdem die Lastberechnung fertig ist. Die Ausführung von Senden und Empfangen eines Telegramms, das hier alle 200 ms gebildet wird, verzögert sich. Die Übertragungszeit wird damit auch beeinflußt und vergrößert sich mit der steigenden Last.

Wegen der breiten Streuung der Übertragungszeit, die durch eine zeitliche Verzögerung der Last entstanden ist, vergrößert sich auch die Varianz. Aufgrund der Asynchronität zwischen den Kommunikationspartnern kann sich bei der Übertragungszeit und bei der Varianz ein lokales Minimum oder Maximum bilden.

Es gibt kaum einen Einfluß auf die Fehlerrate. Die notwendige Rechenzeit für die Kommunikation ist so kurz, daß es nach der Unterbrechung durch höherpriore Anwendungen immer genügende Rechenzeit dafür gibt.

6 Blockorientierte Übertragung

Mit expliziter Verwendung spezifischer Kommunikationsbausteine ist der Anwender in der Lage, eine Kommunikationsverbindung schnell einzurichten. Die Dienste, die unterhalb der Anwendungsschicht zu machen sind, bleiben verdeckt. Dem Anwender wird dabei nur eine Schnittstelle angeboten, womit die Parameter der Kommunikationsbausteine festgelegt werden können.

Man sieht dabei auch gleichzeitig den Nachteil: Für jede Kommunikationsbeziehung muß immer ein Kommunikationsbaustein in den beteiligten Automatisierungssystemen plaziert werden. Falls es einen großen Bedarf an Signalübertragung gibt, bedeutet die Plazierung und Einstellung von Kommunikationsbausteinen einen großen Projektierungsaufwand.

Wegen zunehmender Dezentralisierung von Automatisierungskomponenten ist es sehr wünschenswert, die projektierten Teilpläne bzw. Funktionsteile z.B. aus Gründen der Lastverteilung ohne großen Aufwand von einem in eine andere CPU verschieben zu können. Durch Kommunikationsbausteine, die innerhalb des Plans hierfür plaziert werden müssen, wird dies erschwert, weil alle betroffenen Verbindungen komplett neu projektiert werden müßten. Das bedeutet

- Löschen von alten Kommunikationsbausteinen,
- Einfügen von neuen Kommunikationsbausteinen,
- Parametereinstellung in den neuen Kommunikationsbausteinen.

Der Projektierungsaufwand für eine neue Verteilung von Funktionen auf mehrere Automatisierungssysteme ist somit sehr hoch.

Aus den obengenannten Gründen soll möglichst versucht werden, auf die Plazierung von Kommunikationsbaustein zu verzichten. Dem Anwender bleibt die zeitaufwendige Einstellung des Parameters (z.B. ID, R_ID) damit erspart. Man geht zu einer verdeckten, impliziten Variante der Kommunikation mit **blockorientierter Übertragung** über.

6.1 Übersicht

Eine blockorientierte Übertragung sieht vor, alle zwischen zwei AS auszutauschenden Prozeßdaten zuerst in einem Puffer zwischenzuspeichern und innerhalb bestimmter Zykluszeiten nicht einzeln, sondern in Blöcken zusammengefaßt zu übertragen.

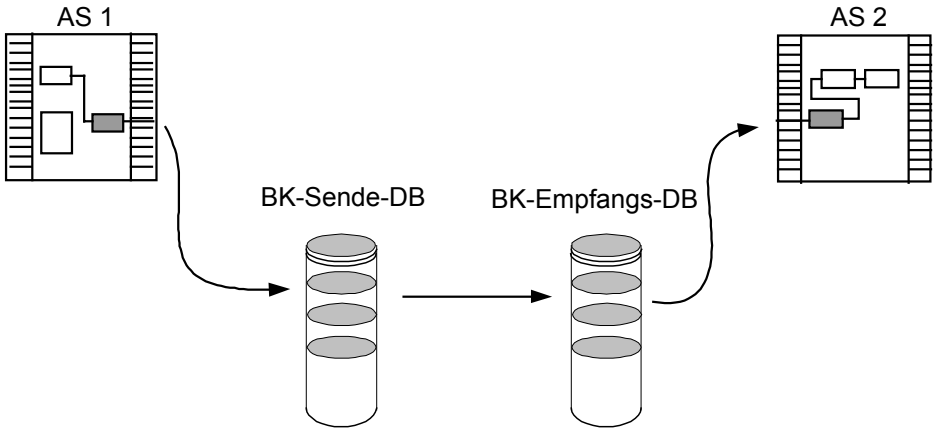


Abbildung 6.1: Prinzipdarstellung

Dazu werden in beiden Automatisierungssysteme bei der Codegenerierung spezielle DBs als Zwischenpuffer für die blockorientierte Übertragung generiert (BK-DBs) (Abbildung 6.1). Diese BK-DBs haben den Charakter von Sende- bzw. Empfangspuffern und besitzen auf der Sende- und Empfangsseite eine identische Struktur. Die auszutauschenden Parameter werden jeweils vor dem Senden aus den technologischen Bausteinen gelesen und in den BK-Sende-DB kopiert. Anschließend wird dieser BK-DB übertragen. Auf der Empfangsseite werden die Parameter nach dem Empfang aus dem BK-Empfangs-DB in die technologischen Bausteine kopiert.

6.2 Ablauf der blockorientierten Übertragung

6.2.1 Sendeseite

Die BK-DBs werden bei der Codegenerierung auf reale Datenbausteine der Ressourcen abgebildet. Damit auch die Datenübertragung zwischen den Ressourcen zum Ablauf kommt, müssen zusätzlich Funktionen generiert und zyklisch angestoßen werden, die den Datentransport zwischen Sender-BK-DB und Sendebaustein leisten.

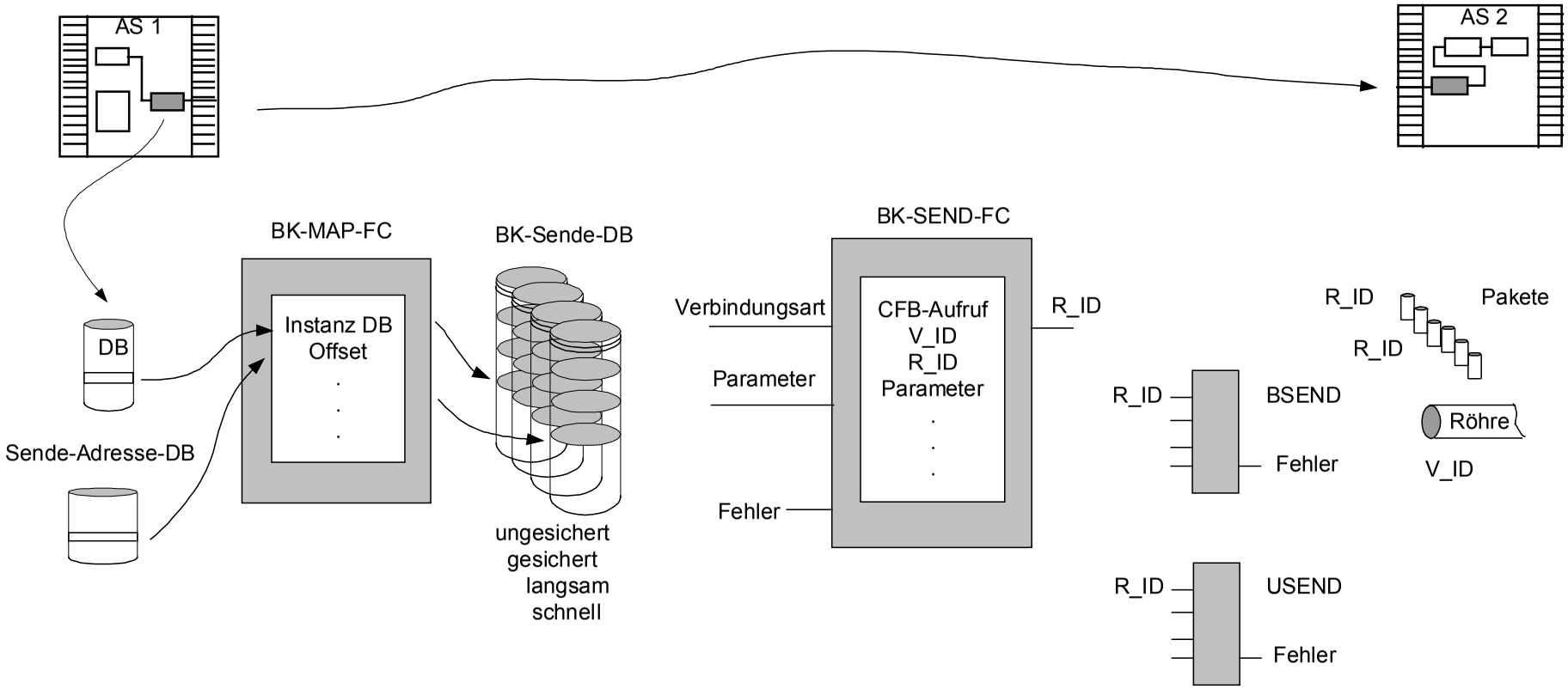


Abbildung 6.2: Ablauf auf der Sendeseite

In der Abbildung 6.2 transportiert die BK-MAP-FC den Prozeßwert des Bausteinausgangs von der Quelle aus dem DB in das zugehörige Datenelement der BK-DBs. Die Adresseninformation, die den Bausteinanschluß eindeutig identifiziert, wird im Sende-Adresse-DB gespeichert. Die BK-MAP-FC wird ausschließlich sendeseitig erzeugt. Der Aufruf der BK-MAP-FC muß vor dem eigentlichen Senden der Daten erfolgen, damit gewährleistet ist, daß aktuelle Daten übertragen werden.

Für jedes zu transportierenden Datenpaket kann eins von insgesamt vier Möglichkeiten zum Einsatz kommen. Folgende Verbindungstypen sind vorgesehen:

- schnell gesichert
- langsam gesichert
- schnell ungesichert
- langsam ungesichert

Die Blockübertragung läuft zyklisch. Langsam bzw. schnell bezieht sich auf die Zykluszeit (langsam, z.B. default =1 s, schnell, z.B. default = 0,1 s), mit der die Daten zur Laufzeit übertragen werden. Diese Zyklen stellen keine Systemgrenze vom Konzept her dar. Auch kleinere Zyklen sind möglich. Hier ist letztendlich die Leistungsfähigkeit der verwendeten CPU ausschlaggebend.

Die Angaben zu gesichert bzw. ungesichert beziehen sich auf den Kommunikationsdienst. Bei gesichert wird BSEND/BRCV verwendet, bei ungesichert USEND/URCV. Hier optimiert der Anwender die Attribute an der Verbindung, indem er z.B. eine Verbindung von "langsam" nach "schnell" ändert. Auch die gesicherte bzw. nicht gesicherte Übertragung hat Auswirkungen auf die Performance, da der USEND ca. 25% schneller ist als der BSEND. Pro Verbindungstyp und Partner-Automatisierungssystem wird jeweils ein eigener BK-Sende-DB und BK-Empfangs-DB angelegt.

Die BK-Send-FC leistet den entsprechenden Aufruf an die CFB. Dabei werden die benötigten IDs für die Verbindungen erfragt und die CFB damit intern parametriert. Die Zuordnung zwischen Sender-CFB und Empfänger-CFB wird über die identische Request-ID (R_ID) hergestellt.

6.2.2 Empfängerseite

Der Empfangsablauf wird in der Abbildung 6.3 dargestellt.

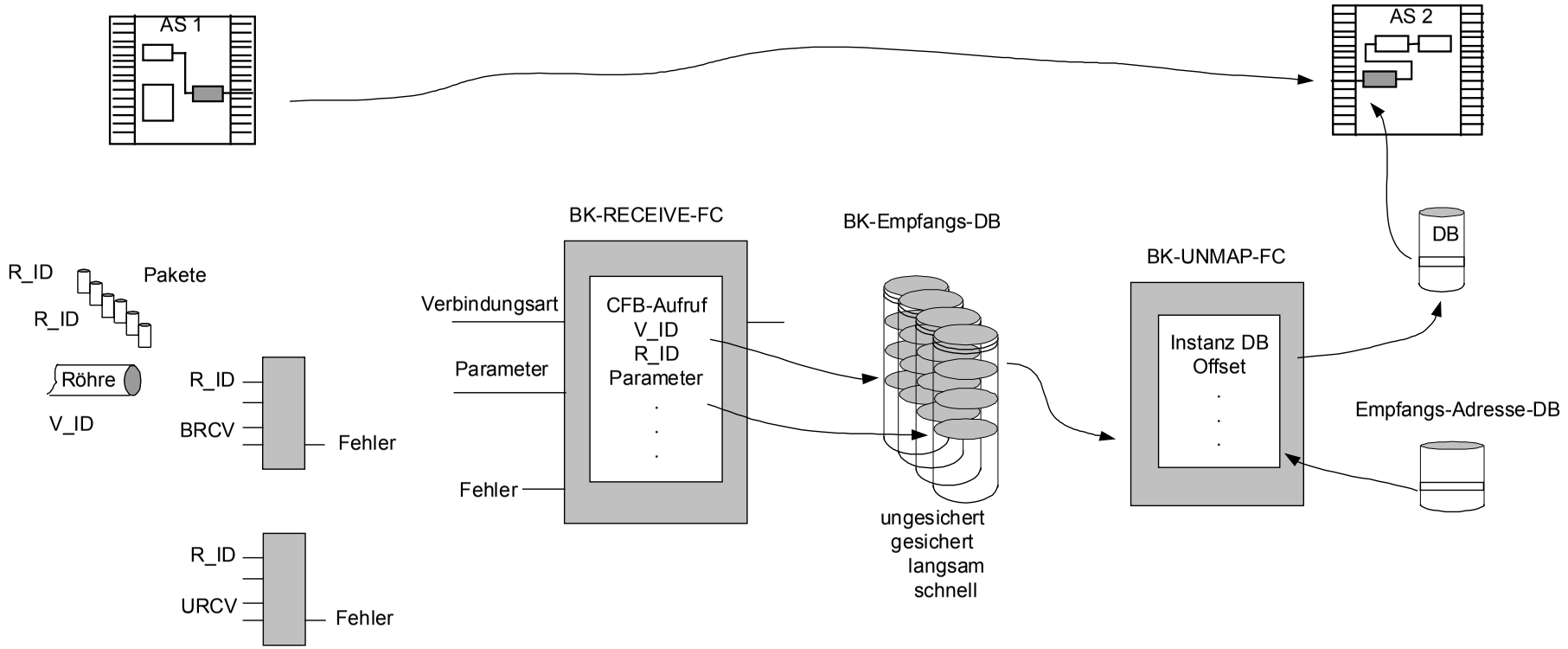


Abbildung 6.3: Ablauf auf der Empfängerseite

Entsprechend der BK-Send-FC wird auf der Empfangsseite die BK-Receive-FC verwendet. Diese Funktion hat die Aufgabe, den Empfänger-CFB zyklisch aufzurufen. Anhand ID und R_ID werden die ankommenden Daten in zugeordnete BK-Empfangs-DBs geschrieben.

Die BK-UnMAP-FC enthält die 'inverse' Funktionalität zur BK-MAP-FC, d.h. mit Hilfe der BK-UnMAP-FC werden Datenelemente aus dem BK-Empfangs-DB in die DBs der Bausteinanschlüsse übertragen. Die notwendige Adresseninformation steht im Empfangs-Adresse-DB zur Verfügung. Die BK-UnMAP-FC wird ausschließlich empfangsseitig erzeugt. Der Aufruf der BK-UnMAP-FC muß nach dem eigentlichen Empfangen der Daten erfolgen, damit gewährleistet ist, daß die aktuellen Empfangsdaten transportiert werden.

6.2.3 Zeitgesteuerter Aufruf der BK-FCs

Die BK-FCs (BK-Send-FC, BK-MAP-FC, BK-Receive-FC, BK-UnMAP-FC) werden sende- und empfangsseitig in einem projektierten Zyklus aufgerufen. Um eine Entkopplung von der OB-Programmierung zu erreichen, werden die BK-FCs aus einer gemeinsamen FC aufgerufen. Die Bearbeitung von Kommunikationsbausteinen kann dann nicht durch andere Kommunikationsbausteine unterbrochen werden.

Zwischen Aufruf-FC und BK-FCs wird ein Zeitregister zwischengeschaltet, mit deren Hilfe auch entsprechende Untersetzungen (langsam, schnell) realisiert werden können.

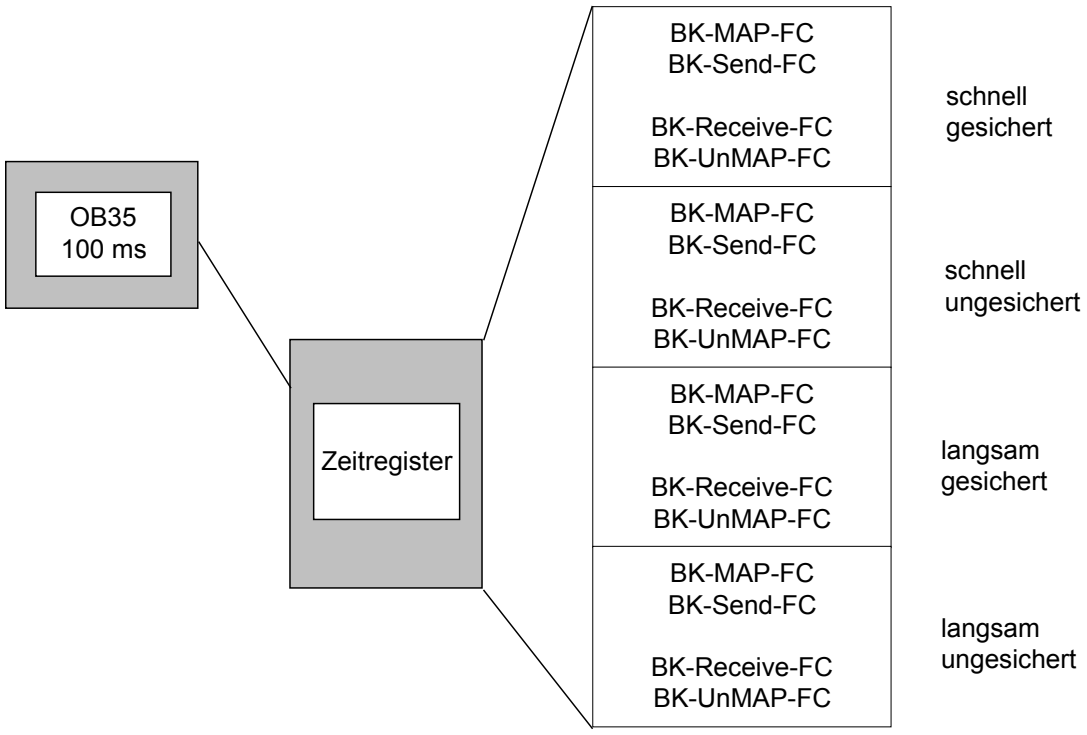


Abbildung 6.4: Zeitgesteuerter Aufruf der BK-FCs

6.3 Codegenerierung

6.3.1 Automatische Generierung von Adresse-DB

Die Adresseninformation darüber, woher die Daten herkommen und wohin die Daten gehen, wird im Adresse-DB gespeichert. Der Adresse-DB wird bei der Codegenerierung erzeugt. Auf der Sendeseite findet man den Sende-Adresse-DB und auf der Empfangsseite den Empfangs-Adresse-DB.

Der Compiler überprüft die projektierten Verbindungen, d.h. die geräteübergreifenden Verschaltungen, nach der Reihenfolge. Die Adresse des jeweiligen Sende-Anschlusses auf einer Seite der Verschaltung wird in den Sende-Adresse-DB eingetragen. Im Empfangs-Adresse-DB wird dann die Adresse des Empfangs-Anschlusses auf der anderen Seite der Verschaltung eingeschrieben (Abbildung 6.5).

Nach der Eintragung aller projektierter Verbindungen ist der Adresse-DB in das AS zu laden. In Betrieb liest die BK-MAP-FC bzw. BK-UNMAP-FC die Adresseninformation, um die Daten richtig zu adressieren.

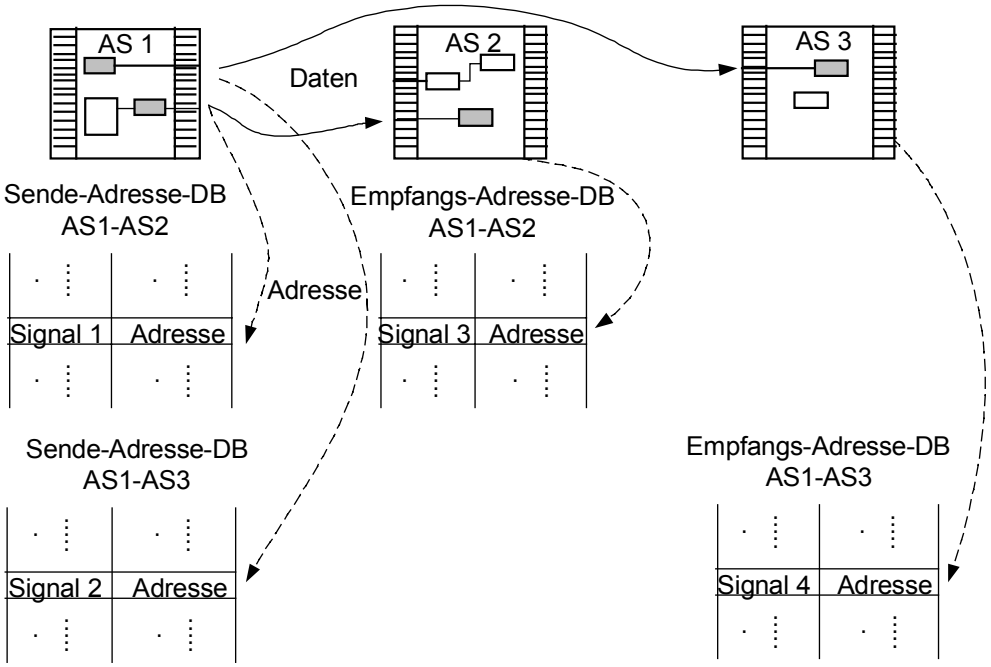


Abbildung 6.5: Generierung von Adresse-DB

6.3.2 Invariante Positionsbelegung in Adresse-DB und BK-DB

Durch die Codegenerierung für die blockorientierte Übertragung entstehen Abhängigkeiten zwischen den beteiligten Automatisierungssysteme, da sowohl auf der Sendeseite als auch

auf der Empfangsseite die Adresseninformation zu generieren ist. Das Konzept sieht hierzu Mechanismen vor, die dennoch eine unabhängige Codegenerierung ermöglichen.

Um diese Unabhängigkeit zu erreichen, muß die Signalbelegung innerhalb eines Sende-Adresse-DB bzw. Empfangs-Adresse-DB invariant gegenüber Änderungsgenerierläufen sein, d.h. bei einer erneuten Codegenerierung, insbesondere bei einer Delta-Generierung dürfen 'einmal vergebene' Adressen nicht geändert werden. Die Position innerhalb des Adresse-DB bzw. BK-DB bei beiden Kommunikationspartnern entsprechen einer eindeutigen Verbindung. Wäre dies nicht gewährleistet, so müßten auch alle schon geladenen Partner-BK-Ressourcen neu generiert und geladen werden.

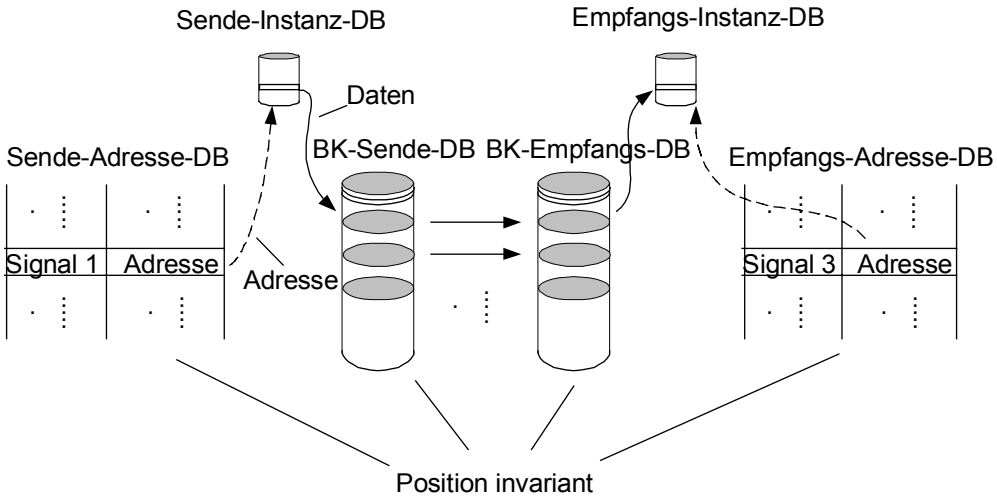


Abbildung 6.6: Invariante Pufferbelegung

Erst mit der invarianten Belegung kann man folgende Kriterien erfüllen:

- Die Codegenerierung auf einzelnen AS kann unabhängig voneinander vorgenommen werden. Die Kommunikationsbeziehung bleibt bestehen. Eine Entkoppelung der Codegenerierung zwischen AS wird erreicht.
- Außerdem hat die invariante Pufferbelegung auch zur Folge, daß die Inbetriebnahme von AS unabhängig voneinander erfolgen können, d.h. ein Kommunikationsteilnehmer kann unabhängig von Kommunikationspartnern weiter projektiert, codegeneriert, geladen und in Betrieb genommen werden.
- Wegen der eindeutigen Zuordnung innerhalb der Adresse-DBs von Sender und Empfänger in jedem Betriebsvorgang wird ein stoßfreies, feingranulares Online-Delta-Laden im prozeßaktiven Betrieb garantiert.

6.3.3 Reorganisation der Adresse-DBs bei der Änderungsprojektierung

Das Anstreben nach Unabhängigkeit zwischen den Kommunikationsteilnehmern bringt einen Nachteil bei der Änderungsprojektierug mit sich. Eine Änderungsprojektierung, die die Kommunikation betrifft, kann durch folgende Vorgänge hervorgerufen werden:

- durch neues Hinzufügen, Löschen oder Umverdrahten einer Verbindung oder
- durch Kopieren, Verschieben oder Löschen von Bausteinen mit geräteübergreifenden Verschaltungen.

Die Projektierungsänderung durch Hinzufügen oder Löschen von den betroffenen Datenelementen wird in den Adresse-DBs verwaltet. Beim Löschen von Verschaltungen werden im Adresse-DB Datenelemente deaktiviert. Entsprechendes gilt auch für das Verschieben von Bausteinen. Die Datenelemente dürfen erst wieder neu belegt werden, wenn sichergestellt ist, daß sowohl die Sende-DB als auch die Empfangs-DB komplett neu generiert und geladen wurden.

Zustandsverwaltung eines Datenelements

Um die entsprechenden DBs nicht in allen Fällen bei einer Änderungsprojektierung komplett generieren zu müssen, wird ein Zusatzmechanismus eingeführt. Hierbei ist insbesondere zu beachten, daß ein Laden der von einer Kommunikationsänderung betroffenen Automatisierungsgeräte grundsätzlich nicht gleichzeitig, sondern immer zeitversetzt erfolgt. Aus diesem Grund ist eine Zustandsverwaltung für freigewordene Datenelemente notwendig.

Zwei Zustandseigenschaften werden hier eingeführt:

- die Zustandseigenschaft ‘belegt' bzw. ‘nicht belegt' deutet an, ob die Adreßinformation für eine Verschaltung im Adresse-DB auf der Sendeseite und der Empfängerseite schon gespeichert ist oder nicht.
- die Zustandseigenschaft ‘geladen’ bzw. ‘nicht geladen’ zeigt, daß der Adresse-DB mit dem entsprechenden Datenelement sowohl in der Sender- und Empfänger-Ressource entweder geladen oder nicht geladen ist.

Durch die Kombination von diesen zwei Zustandseigenschaften ergeben sich 4 mögliche Zustände (Abbildung 6.7):

Zustand ‘nicht belegt' und ’nicht geladen'

Ein Speicherelement im Adresse-DB ist nicht belegt und steht damit frei verfügbar für eine BK-Verschaltung. Bei einer Gesamtgenerierung können alle Datenelemente in den Adresse-DBs in diesen Zustand versetzt werden, damit ein lückenloser Neuaufbau der Adresse-DBs erfolgen kann.

Zustand ‘belegt’ und ‘nicht geladen’

Ein Speicherelement im Adresse-DB ist belegt, aber noch nicht geladen, z.B. Zustand direkt nach der Übersetzung der Änderungsprojektierung. Es findet keine Kommunikation des Signals statt.

Zustand ‘belegt’ und ‘geladen’ (Aktiver Modus)

Ein Speicherelement im Adresse-DB ist belegt und auch geladen, z.B. Zustand direkt nach dem Laden der Änderungsprojektierung. Es findet hier die Kommunikation schon statt.

Zustand ‘nicht belegt’ und ‘geladen’

Speicherelement im Adresse-DB ist nicht belegt, aber vorher schon geladen (alte Projektierungsdaten). Durch eine Reorganisation und beidseitiges Neuladen kann dieses Element wieder nutzbar gemacht werden.

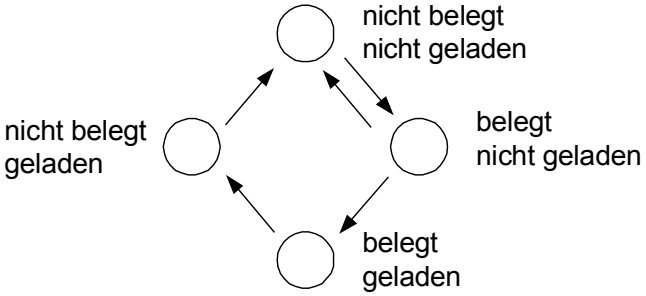


Abbildung 6.7: Zustandsverwaltung für ein Datenelement

Nach der Generierung von Adresse-DBs haben alle Speicherelemente einen Zustand 'nicht belegt' und 'nicht geladen'. Durch einen Generiervorgang erfolgt dann ein Zustandswechsel von 'nicht belegt' und 'nicht geladen' nach 'belegt' und 'nicht geladen', wobei die umgekehrte Richtung für eine mehrfache Änderungsprojektierung und Generierung gilt. Nach den beiderseitigen Ladevorgängen wird wiederum der Zustand 'belegt' und 'geladen' erreicht. Das Löschen oder die Umschaltung einer Verbindung und ein danach folgender Generiervorgang bewirkt einen Zustandswechsel von 'belegt' und 'geladen' nach 'nicht belegt' und 'geladen'. In dem anschließenden Ladevorgang wird das freigewordene Speicherelement mit dem Zustand 'nicht belegt' und 'nicht geladen' markiert.

Verbindungsgranulare Reorganisation

Es besteht auch eine andere Möglichkeit, die während der Änderungsprojektierung in den Adresse-DBs entstehenden Lücken sinnvoll zu verwalten. Man kann für je eine Kommunikationsbeziehung zwischen zwei Ressourcen eine Reorganisation anstoßen. Damit werden die Adresse-DBs (bezogen auf diese Beziehung) komplett neu aufgebaut, wobei die Datenelemente lückenlos aneinander geschoben werden. Danach sind beide Ressourcen neu zu laden. Dabei werden mindestens die Adresse-DBs neu generiert.

Wesentlich ist, daß die Reorganisation verbindungsgranular möglich sein muß, da sonst eine Neugenerierung des Gesamtprojekts notwendig wäre.

6.3.4 Hilfsinformation für den Codegenerator

Für die Datenübertragung werden zuerst zwei Kommunikationsdienste BSEND/BRCV und USEND/URCV angeboten. Aber für verschiedene Zielsysteme ist auch der Einsatz von unterschiedlichen Kommunikationsmechanismen mit unterschiedlichen Eigenschaften notwendig. Die Festlegung der eingesetzten Kommunikationsmethode kann durch den Anwender eingestellt werden oder automatisch erfolgen.

Bei automatischer Einstellung sollte diese Information im Codegenerator nicht einprogrammiert sein, sondern ihm per Hilfsinformation zur Verfügung gestellt werden. Damit soll erreicht werden, daß der Codegenerator weitestgehend unabhängig von den verwendeten Kommunikationsmechanismen entwickelt werden kann.

Tabelle 6.1 zeigt andeutungsweise die zwischen den Zielsystemen zu verwendenden Dienste.

	S7-400	S7-300
S7-400	BSEND/USEND	XSEND
S7-300	XSEND	XSEND

Tabelle 6.1: Mögliche Kommunikationsdienste zwischen unterschiedlichen Zielsystemen

Nach dem Auswählen des eingesetzten Kommunikationsdienstes sollen noch die Parameter vom jeweiligen Dienst bekannt gemacht werden, damit z.B. die maximale Blocklänge für die Übertragung ermittelt werden kann. Diese Information kann auch in Form von Tabellen intern zur Verfügung gestellt werden.

Tabelle 6.2 beschreibt einige Eigenschaften der Dienste, die für die Codegenerierung relevant sind:

	BSEND/BRCV	USEND/URCV	XSEND/XRCV
Blocklänge	64 K	440 Byte	76 Byte
Parameter	V_ID R_ID	V_ID R_ID	V_ID R_ID

Tabelle 6.2: Parameter der Kommunikationsdienste

6.4 Überwachung und Fehlerbehandlung

Blockorientierte Übertragung übernimmt die manuelle Projektierung der Kommunikationsbausteine und Verbindungen intern nahezu vollständig. Der Anwender muß sich nicht mehr mit Verbindungsnummer oder Request_IDs befassen. Diese Projektierungsdaten werden automatisch durch einen internen Mechanismus erzeugt.

Dennoch kann nicht ausgeschlossen werden, daß im laufenden Betrieb oder während der Test- und IBS-Phase Übertragungsfehler auftreten. Übertragungsfehler können unterschiedliche Ursachen haben.

Mögliche Ursachen können Projektierungsfehler oder Hardwarestörungen sein. Dem Anwender soll deshalb eine Methode zur Verfügung gestellt werden, welches im Störungsfall eine komfortable und einfache Fehlerdiagnose ermöglicht und darüber hinaus eine für die IBS-Phase geeignete Unterstützung bietet. Darüber hinaus müssen bei Störungen die Fehler- und Statusmeldungen der CFBs auswertbar sein.

Im Fehlerfall soll nach Möglichkeit vermieden werden, daß der reguläre Betrieb der AS durch falsch oder nicht angekommene Daten negativ beeinflußt wird. Eine Umschaltung auf einen sicheren Betrieb soll zur jeder Zeit möglich sein.

6.4.1 Überwachungsfunktion

Eine wesentliche Anforderung an eine Test- und IBS-Unterstützung ist es, die Ursache von Fehlern und Störungen im laufenden Betrieb erkennen und über ein Online-Delta-Laden beheben zu können.

Sporadische Störungen können in der Regel nur im laufenden Betrieb festgestellt werden. Aus diesem Grund sind Überwachungsfunktionen erforderlich, die in den laufenden Betrieb eingebunden werden. Ziel ist es, Signale zum Zeitpunkt des Sendens oder des Empfangs aufzuzeichnen. Überwachungsfunktionen sind laufzeitoptimal zu realisieren, damit das Zeitverhalten der CPU idealerweise nicht beeinflußt wird.

Im Test- und IBS-Modus können über die CFC/SFC CPU-übergreifende Verschaltungen oder Verbindungen ausgewählt und beobachtet werden. Für die ausgewählten Verschaltungen wird ein Funktionsbaustein erzeugt: BK-Moni-FC. Mit Hilfe dieses Funktionsbausteines wird der Zustand (Wert) der Verschaltung aus den Sende- oder Empfangspuffern gelesen und mit Hilfe von Rangieranweisungen in einen Datenbaustein (BK-Moni-DB) geschrieben (Abbildung 6.8).

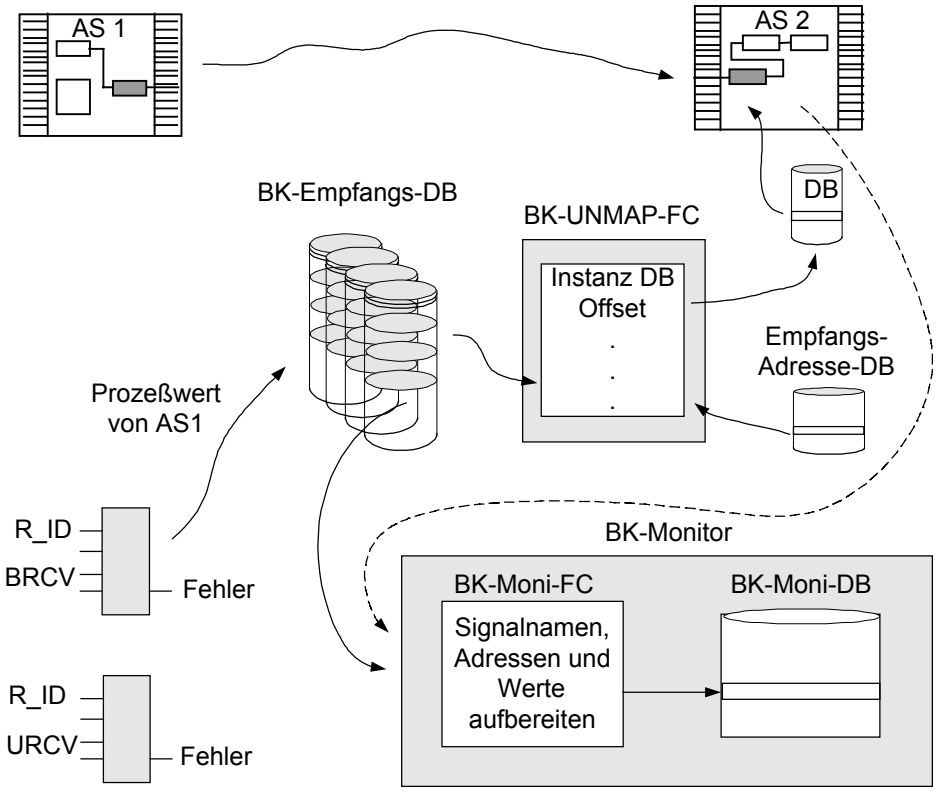


Abbildung 6.8: Überwachungsfunktionen

Durch Auslesen des BK-Moni-DB können die CPU-übergreifenden Verschaltungen beobachtet werden. Im laufenden Betrieb können sowohl System- als auch Projektierungsfehler entdeckt und anschließend analysiert werden. Zusätzlich zum Wert der Signale sollen auch Status- und Fehlerzustände der Kommunikationsbausteine signalgranular überwacht werden können.

Die Aktivierung der Überwachungsfunktionen erfolgt im laufenden Betrieb. Mit Hilfe der Überwachungsfunktionen werden die Sende- und Empfangspuffer beobachtet. Es ist möglich den gesamten Puffer oder aber den Status einzelner Signale zu beobachten. Voraussetzung für das Aktivieren von Überwachungsfunktionen ist jedoch, daß die Übertragung generell arbeitet.

6.4.2 Ersatzwert und Defaultwert

Um einen Prozeß immer in einem sicheren Zustand halten zu können, ist eine Ersatzwert-Strategie notwendig. Ersatzwerte werden nur dann benutzt, wenn das AS sich in einer der folgenden Zustände befindet:

- Störungsfall (z.B. unterbrochene Verbindung)
- Anlaufphase
- Test und IBS

Darüber hinaus besteht die Anforderung, den gesamten Eingangspuffer definiert vorbelegen zu können (Defaultwert). Hierdurch wird ein definiertes Verhalten der Anwendersoftware ermöglicht.

Mit Hilfe von der CFC oder der SFC werden einzelne Verschaltungen selektiert. Für jede Verschaltung kann ein Ersatzwert (als Attribut) über Masken hinterlegt werden. Mit Hilfe eines Mapping-Vorganges wird für die Ersatzwerte ein Datenbaustein (BK-Force-DB) angelegt. Die Ersatzwerte werden in diesen Datenbaustein hinterlegt.

Falls bei der Übertragung z.B. durch Bewertung von Fehler- oder Statusmeldungen der CFBs Fehler erkannt werden, ist das AS in der Lage, die Ersatzwerte automatisch oder manuell einzusetzen. Das AS übergibt dem Funktionsbaustein (BK-Force-FC) die Information (ID, R_ID), bei der die fehlerhafte Verbindung auftritt. Anhand dieser Information bekommt der BK-Force-FC von BK-RECEIVE-FC eine Adresse im FIFO (BK-Empfangs-DB), zu der die fehlerhaften Daten transportiert werden sollen. Der Ersatzwert wird vom Datenbaustein (BK-Force-DB) geholt und in die gewünschte Position in das FIFO geschrieben. Nicht benutzte Zellen in den BK-Empfangs-DB werden grundsätzlich mit dem Ersatzwert ‘Null’ vorbelegt (Abbildung 6.9).

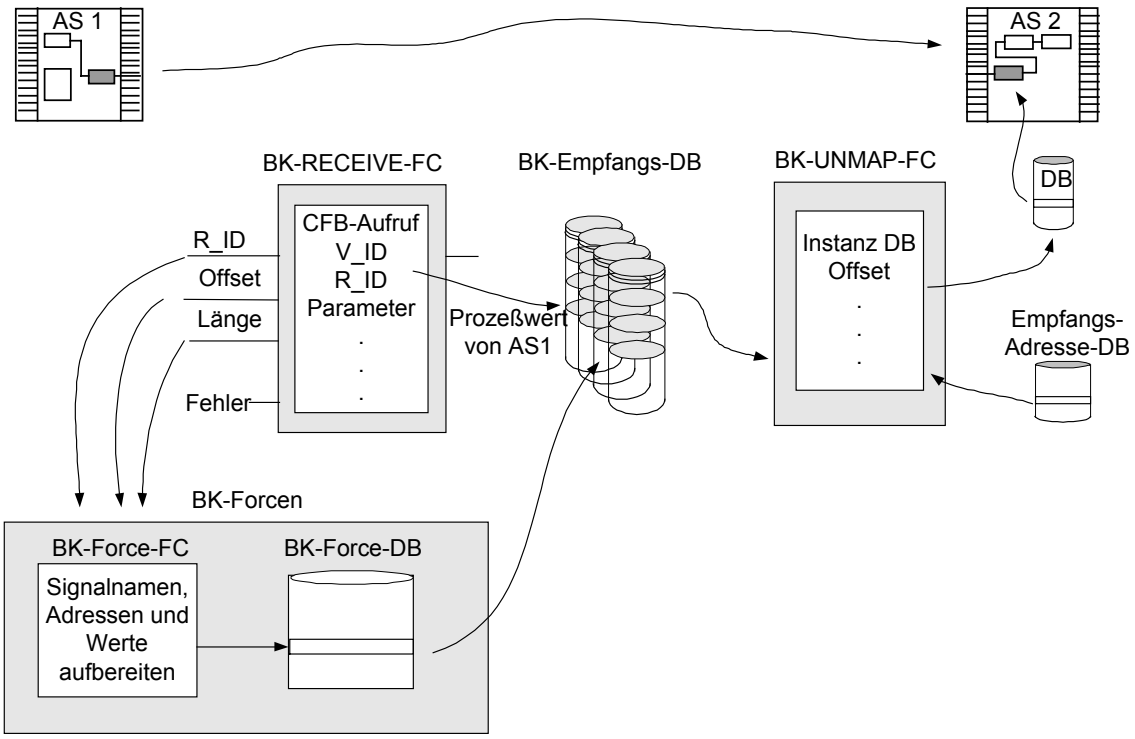


Abbildung 6.9: Ersatzwerte vorgeben

Für den Fall, daß der Sender noch nicht vorhanden ist (z.B. Entscheidung durch Fehler- bzw. Statusmeldung des CFB), muß es möglich sein, auf der Empfängerseite Defaultwerte anzugeben, die auf der Empfängerseite in den BK-DB geschrieben werden. Die Defaultwerte werden an den Eingängen der Bausteine hinterlegt und an Stelle der Prozeßwerte verwendet, wenn diese nicht verfügbar sind.

Mit Hilfe dieser Strategie bleiben die Prozeßwerte innerhalb des BK-Empfangs-DB immer definiert, auch im Fehlerfall, bei Verbindungsausfall oder bei Anlauf bzw. Wiederanlauf. Deshalb kann ein sicherer Betrieb garantiert werden.

6.4.3 Fehlerreaktion

Als Standardreaktion auf Fehlverhalten wird der Fehler gemeldet. Dies umfaßt sowohl die Fehler, die vom AS-Betriebssystem über OB bearbeitet werden, als auch solche, die von den Kommunikationsbausteinen über Fehlerausgänge bzw. Returnvalues gemeldet werden:

- Meldung bei CP Ausfall,
- Meldung bei Verbindungsausfall,
- Darstellung von Fehlermeldungen der CFB-Bausteine.

Die Methode sieht vor, daß für diese Standardfälle ein Code generiert wird, der entsprechend vordefinierte Systemmeldungen an das OS absetzt (vom Prinzip her wie beim Leittechnikmeldekonzept der Baugruppentreiberbausteine). Wichtig ist hierbei, daß gemeldet

wird, zwischen welchen CPUs die Kommunikation gestört ist. Um dies melden zu können, wird eine Information über die CPU und die Verbindung (langsam/gesichert, langsam/ungesichert, ...) im jeweiligen BK-DB abgelegt und über Begleitwerte des ALARM_8P an das OS weitergegeben.

Alle CFBs verfügen über Fehlerausgänge. Bei einem Auftreten von Übertragungsfehlern werden diese Ausgänge aktiviert. Im Fehlerfall sind dann alle über diese Verbindungen bzw. über diese CFBs oder CPs transportierten CPU-CPU-Verschaltungen ungültig und müssen als fehlerhaft im CFC- und SFC-Plan gekennzeichnet werden.

Grundsätzlich muß es möglich sein, Verbindungen zu überwachen und im Fehlerfall definiert auf Fehlermeldungen zu reagieren. Für die Behandlung von Fehlern gibt es mehrere Lösungswege.

Eine Alternative ist das Offenlegen der Mappingergebnisse in BK-DB für die generierten Kommunikationsbausteine. Der Anwender hat damit die Möglichkeit unter Verwendung der verfügbaren Programmier- und Projektierwerkzeuge die Statusinformation der Kommunikationsbausteine auszuwerten und entsprechend zu reagieren. Diese Lösung wird aus Aufwandsgründen bevorzugt.

Eine weitere Lösung wird durch das nachfolgende Konzept beschrieben. Mit dieser ist zusätzlich auch eine Reaktion auf zurückliegende Fehler möglich, da im BK-Error-DB die Fehlerhistorie aufgezeichnet ist.

Die Auswertung der Fehlerausgänge der CFBs erfolgt analog zur Auswertung der Statusausgänge. Mit Hilfe eines Funktionsbausteines (BK-Error-FC) wird die Fehlermeldung bzw. der Fehlerstatus in einen Datenbaustein (BK-Error-DB) geschrieben. Dieser Datenbaustein kann dann ausgewertet werden. Der Aufruf der BK-Error-FCs erfolgt in einem Fehler-OB (Abbildung 6.10).

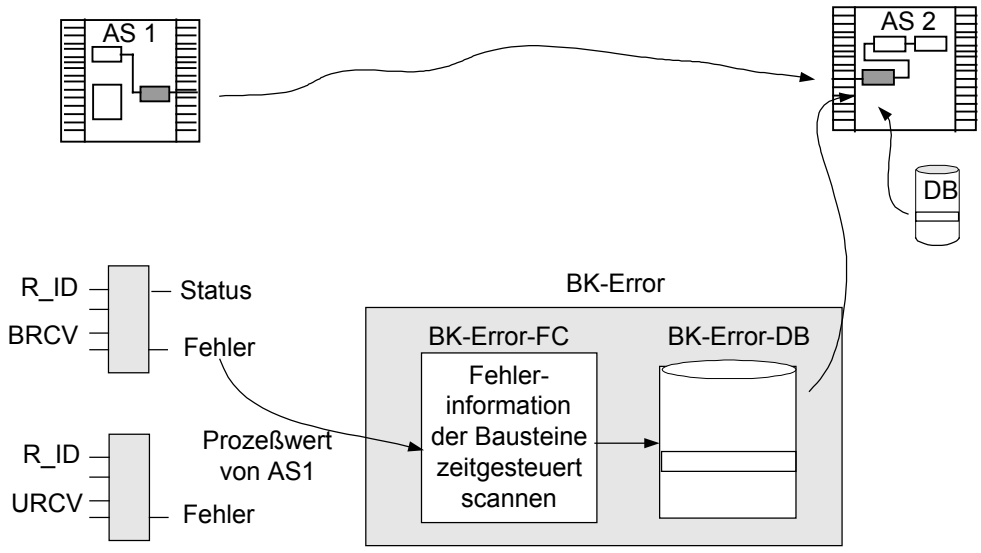


Abbildung 6.10: Reaktion auf Fehler

Eine Alternative zu der oben aufgeführten Lösung wäre, im BK-Error-DB nicht die Historie aufzuzeichnen, sondern dort immer nur den aktuellen Kommunikationszustand pro Puffer zu führen. Dies sollte dann immer in der gleichen Art geschehen, unabhängig davon, welche Art von Kommunikation dahinter steht.

6.5 Meßergebnisse

Für den Test wird ein Prototyp mit Hilfe von Interfacebausteine realisiert, in dem das Modell mit blockorientierter Übertragung zum Tragen kommt. Es werden die Messdaten unter gleichen Bedingungen wie mit der im vorangehenden Kapitel beschriebenen Methode aufgenommen, um die Leistungskenngrößen (Übertragungszeit, Varianz, Fehlerquote) in Abhängigkeit von den Einflußgrößen (Taktzeit, Last) zu untersuchen.

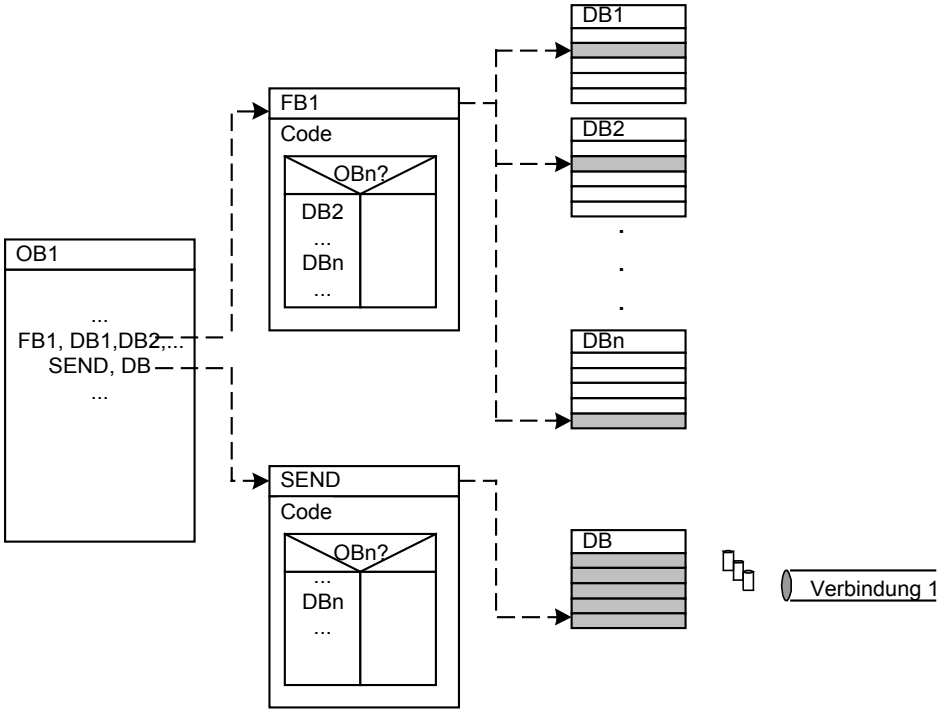


Abbildung 6.11: Bearbeitungsreihenfolge auf der Sendeseite (Blockorientierte Übertragung)

Für den Kommunikationsvorgang werden hier nur zwei Aufrufe benötigt, nämlich Funktionsbaustein für die Organisation der blockorientierte Übertragung und Sendebaustein. Funktionsbaustein verwaltet die zu übertragenden Signale und stellt Sendebaustein die Daten zur Verfügung (Abbildung 6.11). Die Bearbeitungszeit T_{2S} für den Funktionsbaustein ist je nach maximal zulässiger Signalanzahl konstant. Der Zeitaufwand auf der Sende- und Empfängerseite ist,

$$t_{2S} = T_{2S} + T_{SEND}$$

$$t_{2R} = T_{2R} + T_{RCV}$$

Einfluß der Taktzeit

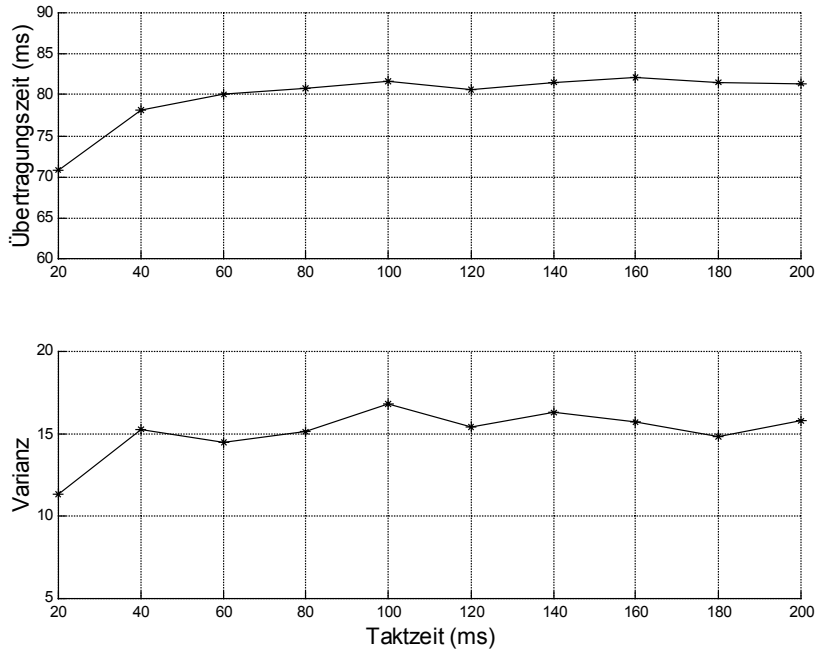


Abbildung 6.12: Übertragungszeit und Varianz in Abhängigkeit von der Taktzeit

Die Übertragungszeit, in der die Daten von einem Automatisierungssystem gesendet und wieder empfangen werden, beträgt hier ebenfalls etwa 80 ms. Die Varianz der Übertragungszeit liegt bei 15 ms^2. Es gibt keinen großen Unterschied im Vergleich zur expliziten Methode, weil die Verarbeitungszeit für die blockorientierte Übertragung auf den Kommunikationspartnern sehr kurz ist. Die Fehlerquote hat auch die gleiche Größenordnung, weil insgesamt der Zeitaufwand für die interne Abwicklung der Kommunikationsaufgabe gleich bleibt.

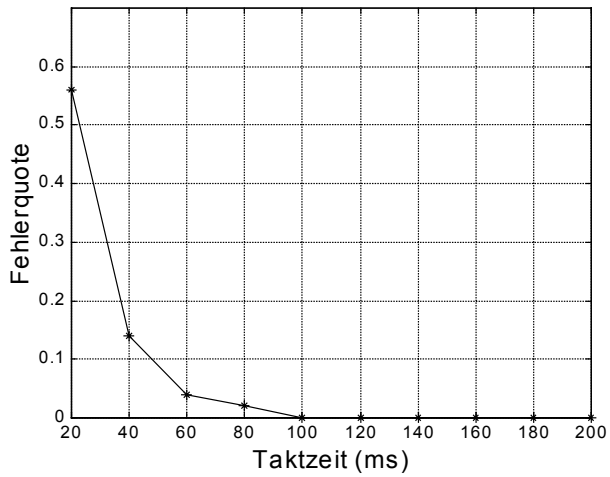


Abbildung 6.13: Fehlerquote in Abhängigkeit von der Taktzeit

Einfluß der Last

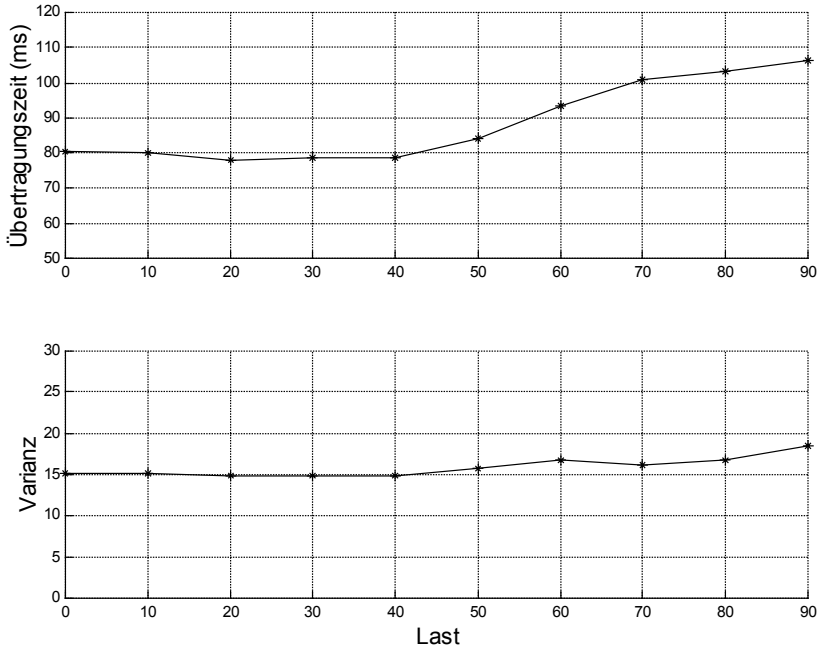


Abbildung 6.14: Übertragungszeit und Varianz in Abhängigkeit von der Last

Mit steigender Last ist die Tendenz von Übertragungszeit und Varianz auch steigend. Da die Daten, die für einen bestimmten Kommunikationsdienst vorgesehen sind, nur einmal diesen Dienst in Anspruch nehmen, ist die Rechenzeit für die Kommunikation sehr kurz. Damit bleibt die Fehlerquote bei blockorientierter Übertragung auf einem sehr niedrigen Niveau.

7 Implizite Kommunikation mit ereignisgesteuerten Mechanismen

Die bisher vorgestellten Konzepte basieren auf einer zyklischen Datenübertragung, wie das Regelkreisbeispiel in der Abbildung 5.13 zeigt. Dieses Übertragungsverfahren ist besonders dort geeignet, wo ein geringes Datenaufkommen vorhanden sind. In Anlagen, in denen ein dichter Datenverkehr innerhalb des Netzes vorhanden ist, wird das Problem auftauchen, ob der Bus immer noch genügend Zeit für die zyklische Übertragung zur Verfügung stellen kann. Um diesen möglichen Verkehrsengpaß verbessern oder vermeiden zu wollen, wird hier eine ereignisgesteuerte Übertragung eingeführt. Das Automatisierungssystem sendet die Daten nur dann, wenn eine Änderung vorliegt. Ein Kommunikationsdienst soll nicht wie bei den bisher betrachteten Methoden in bestimmten OB eingebaut und damit zyklisch aufgerufen werden. Es soll ein Kontrollmechanismus vorhanden sein. Dadurch wird ständig überprüft, ob eingestellte Bedingungen zur Übertragung erfüllt sind oder nicht. Es ist eine Grundlast entstanden, und der Datenverkehr wird dadurch stark reduziert. Eine ereignisgesteuerte Übertragung ist besonders dann geeignet für die Prozesse, in dem sich die Prozeßwerte langsam ändern. In der Abbildung 7.1 werden die unnötigen Übertragungen mit schwarzen Impulsen gekennzeichnet.

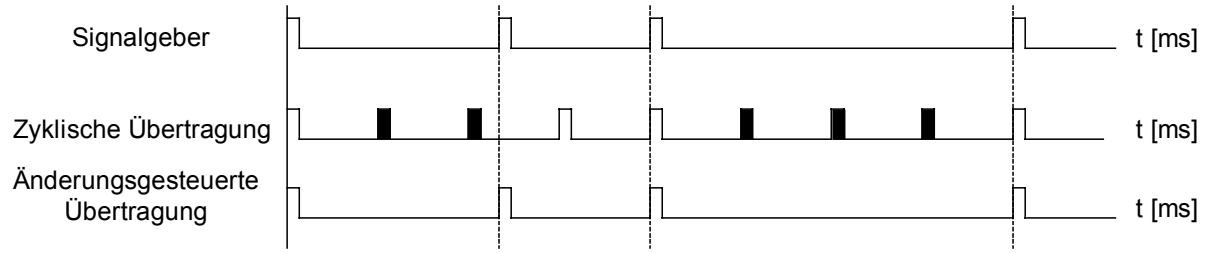


Abbildung 7.1: Unterschied zwischen zyklischem und ereignisgesteuertem Senden

Hinsichtlich einer Dezentralisierung der AS-Aufgaben wird angestrebt, eine Lokalisierung der Information zu erreichen. Das heißt, jedes AS kennt alle Daten in seiner Umgebung. Falls ein AS im Netz ausfällt oder die Verbindung nicht richtig funktioniert, werden andere AS nicht beeinflußt, eine Funktion auszuführen, die nicht das unerreichbare AS betrifft. Damit wird eine höhere Robustheit erreicht.

Eine Information über die in einem AS vorgenommene Änderungsprojektierung soll möglichst nur diesem AS bekannt gemacht werden. Falls die Information auch anderen AS bekannt gemacht werden soll, wird ein Informationsaustausch zwischen beiden Komponenten während der Anlaufphase benötigt (Abbildung 7.2).

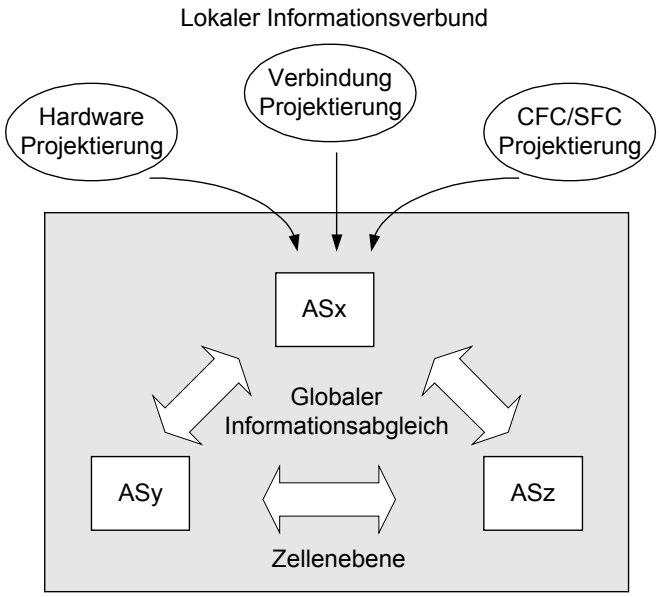


Abbildung 7.2: Lokalisierung der Information

7.1 Übersicht

Die hier vorgestellte Methode, **Implizite Kommunikation mit ereignisgesteuerten Mechanismen**, sieht vor, in dem Sender-Automatisierungssystem die Adreßinformationen für die zu übertragenden Prozeßwerte in eine Tabelle (Fetchstack) abzulegen. Beim Verbindungsaufbau zu dem Partner-Automatisierungssystem werden die korrespondierenden Adressen im Partnergerät aus einer Tabelle (Referenztabelle) entnommen und an das Sender-Automatisierungssystem geschickt. Dort werden sie zu den anderen Steuerinformationen in das Fetchstack gelegt. Durch einen Systemmechanismus wird dieser Befehls-Stack zyklisch abgearbeitet. Bei Erfüllung einiger vorgegebener Kriterien werden der Prozeßwert und die Adreßinformation in ein Telegramm geschrieben und an das Partner-Automatisierungssystem übertragen. Durch einen Systembaustein in dem Partner-Automatisierungssystem wird das Telegramm interpretiert, und anhand der übertragenen Adreßinformation werden die Prozeßwerte an entsprechende Bausteine weitergeleitet (Abbildung 7.3).

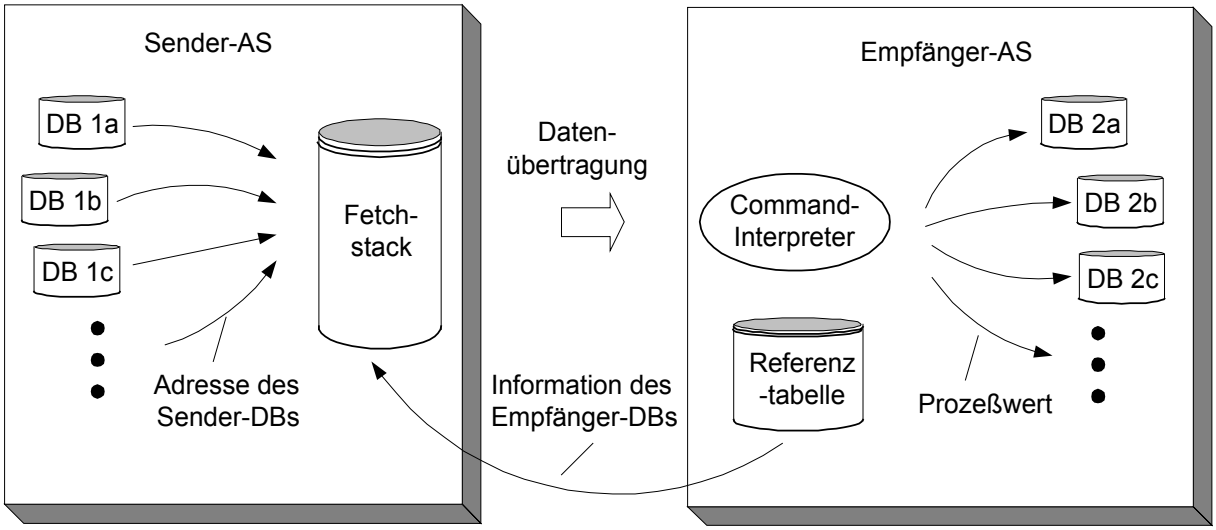


Abbildung 7.3: Implizite Kommunikation mit ereignisgesteuerten Mechanismen

7.2 Sendeseite

7.2.1 Architektur beim Sender

Für jedes Empfänger-Automatisierungssystem existiert ein Fetchstack auf der Sendeseite. Anhand der projektierten Verbindungen wird ein übergeordneter Compiler in dem Übersetzungsvorgang oder eine FC in der Initialisierungsphase das Fetchstack mit der Adresse des Sender-DBs versorgen. Mit dieser Adresse wird der FETCH-FC künftig den Prozeßwert zyklisch aus dem DB holen.

In Fetchstack gibt es auch Speicherzellen, die für das Ziel, nämlich Empfänger-DBs, reserviert sind. Diese Information wird in der Anlaufphase durch ein Sondertelegramm von dem Empfänger-Automatisierungssystem geholt. Folgende Informationen sind darin enthalten:

- Adresse der Empfänger-DBs,
- Datenlänge des zu sendenden Prozeßwertes,
- minimale Wartezeit, die das Sende-Automatisierungssystem zwischen zwei Sendevorgängen warten muß.

Das Sender-Automatisierungssystem hat damit alle Informationen, die zum Datenaustausch über eine geräteübergreifende Verbindung notwendig sind.

Im FETCH-FC wird ein Fetch-Mechanismus eingebaut, durch den die definierten Kriterien kontrolliert werden können. In jedem Zyklus liest der FETCH-FC den aktuellen Wert des im Fetchstack eingetragenen Sendeanschlusses ein und entscheidet, ob die gewünschten Kriterien zum Senden erfüllt sind, z.B. ob eine Änderung der Daten vorliegt.

Falls es zutrifft, werden die aktuellen Daten und die Zieladressen in FIFOs abgelegt. Für die Kommunikation zwischen den Automatisierungssystemen sind nur Daten-FIFOs vorgesehen. FIFOs besitzen eine eigene Priorität:

- Daten-FIFO hoher Priorität: für die Übertragung von höherprioren Prozeßwerten und von Sondertelegrammen, die vom Kommunikationspartner zur Initialisierung oder Adaption benötigt werden.
- Daten-FIFO niedriger Priorität: für niederpriore im Fetchstack stehende Prozeßwerte.

Durch die verschiedenen Prioritäten wird auch im Hochlastfall gewährleistet, daß die wichtigen Daten an das Empfänger-Automatisierungssystem gesendet werden. Die FIFOs sind einmal pro an das Automatisierungssystem angeschlossenes Nachbar-Automatisierungssystem vorhanden.

FIFOs werden auch zyklisch überwacht. Für den Fall, daß die Daten in FIFOs vorhanden sind, wird die Phase der Telegrammbildung eingeleitet. Beim Packen eines Telegramms werden grundsätzlich zuerst die Daten im FIFO mit hoher Priorität mit Hilfe von FC gelesen und in einen Pufferbereich geschrieben. Falls die maximale Länge des Telegramms nicht überschritten ist, werden auch die Daten im FIFO mit niedriger Priorität berücksichtigt. Nachdem alle Daten in den FIFOs ausgelesen sind oder die maximale Länge des Telegramms erreicht wurde, wird das Telegramm fertiggestellt. Ist das Datenaufkommen zu hoch, besteht die Gefahr eines Überlaufs in den niederprioren FIFO.

Das Versenden des Telegramms geschieht durch den systemseitig vorhandenen Kommunikationsbaustein BSEND. Damit ist eine gesicherte Übertragung als Kommunikationsdienst angeboten. Die notwendigen Parameter (z.B. ID, R_ID) werden intern vergeben.

Der Kommunikationsablauf auf der Sendeseite ist in der Abbildung 7.4 aufgezeichnet.

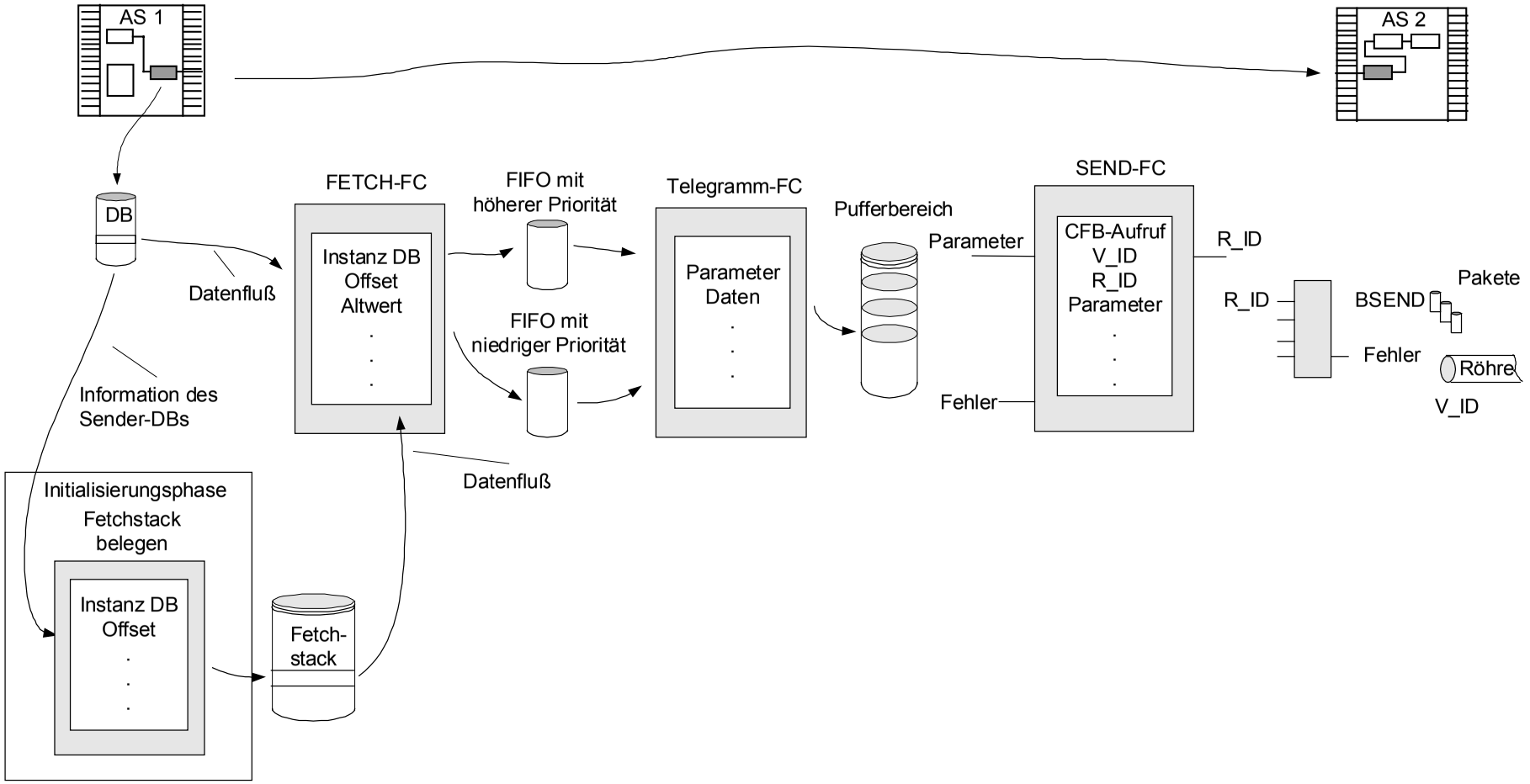


Abbildung 7.4: Kommunikationsablauf in einem Sender-Automatisierungssystem

7.2.2 Eigenschaften des Fetch-Mechanismus

Das erste Modell mit expliziter Projektierung sieht vor, einen Prozeßwert in einem bestimmten Zeitabstand zyklisch zu senden. Der Zyklus wird vom Anwender je nach Bedarf festgelegt. Diese Vorgehensweise ist geeignet für Prozesse, in denen sich die Daten in regelmäßigen Abständen ändern.

In vielen Prozessen ändern sich die relevanten Werte aber zufallsbedingt. Die Signale bleiben zu einem großen Teil der Zeit gleich. Es ist nicht sinnvoll, gleiche Werte in regelmäßigen Abständen zu senden und den Bus unnötig zu belasten. Deshalb wird hier ein neuer Fetch-Mechanismus eingeführt.

Belegung des Fetchstack

Der Fetch-Mechanismus sieht vor, für jeden Kommunikationspartner einen Fetchstack zu reservieren. Informationen über Sendeanschlüsse, deren Werte gesendet werden sollen, werden nach der Reihenfolge in das für das Empfänger-Automatisierungssystem reservierte Fetchstack eingetragen. Der Aufbau des Fetchstack ist in der Abbildung 7.5 aufgezeichnet.

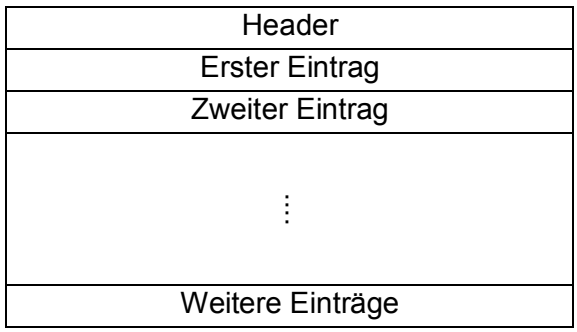


Abbildung 7.5: Fetchbelegung

Die Defaultgröße des Fetchstack kann vom Anwender eingestellt werden und wird zuerst auf 200 Einträge gesetzt. Der Header ist 10 Bytes lang. Dort steht die Information, die für die Organisation des Fetch-Mechanismus notwendig ist:

- Summe der belegten Einträge,
- Position des zuletzt gelesenen Eintrags, usw.

Mit dieser Information ist das Automatisierungssystem in der Lage, alle Einträge im Fetchstack nach der Reihenfolge abzufragen und bei neuen oder gelöschten Einträgen das Fetchstack neu zu organisieren.

Nach dem Header folgen erst die Einträge, die geräteübergreifende Verbindungen beschreiben. Jeder Eintrag ist 18 Bytes lang. In einem Eintrag gibt es zwei Arten von Informationen:

- Adressinformation,
- Kontrollinformation.

Die Adreßinformation definiert einen eindeutigen Kommunikationsweg zwischen Sender- und Empfänger-Baustein. Dazu gehört sowohl die Adresse des Sender-Bausteins, aus dem die Daten kommen, als auch die Adresse des Empfänger-Bausteins, in den die Daten eingeschrieben werden sollen.

Mit Hilfe der Kontrollinformation kann das Sender-Automatisierungssystem entscheiden, ob der Prozeßwert von dem durch die Adreßinformation festgelegten Anschluß des Sender-Bausteins übertragen werden soll.

0	(Destination DB)	
2	(Destination Offset)	
4	S	(Destination Länge)
6	Zähler	(Minimaler Zyklus)
8	Altwert byte 0, 1	
10	Altwert byte 2,3	
12	reserviert	reserviert
14	Source DB	
16	Source Offset	

Abbildung 7.6: Belegung eines Eintrags

In der Abbildung 7.6 wird die Belegung des Eintrags gezeigt. Die ersten 6 Bytes beinhalten die Information über den Empfangs-DB. Die erforderliche Information steht aber in diesem Zeitpunkt noch nicht zur Verfügung. Sie wird später durch ein Sondertelegramm des Empfänger-Automatisierungssystems in der Koordinierungsphase eingetragen.

Im nächsten Feld steht die gewünschte minimale Periode des Fetch-Auftrags. Die Periode wird als ein Vielfaches einer vordefinierten Zeiteinheit (z.B. 1 Sekunde, abhängig von OB) gesetzt. Der Übertragungszyklus kann als Parameter für jeden zu übertragenden Prozeßwert individuell eingestellt werden. Um die Zeit zu überwachen, ist ein Zähler notwendig. Der Zählerstand wird jedesmal beim Durchlaufen des Fetchstack ausgewertet. Das Ergebnis sagt aus, ob dieser Prozeßwert übertragen werden kann oder nicht.

Diese minimal erreichbare Periode ist dazu gedacht, daß nicht jede Änderung sofort übertragen werden soll. In der Praxis kommt es öfter vor, daß die Strecke spontan gestört oder durch zeitweises Rauschen beeinflußt wird. Die dadurch entstehende Änderung beinhaltet wenige Informationen über die reale Strecke und soll in den meisten Fällen nicht berücksichtigt werden. Die Übertragung solcher Änderungen würde das Automatisierungssystem stark belasten und den Kommunikationsweg blockieren.

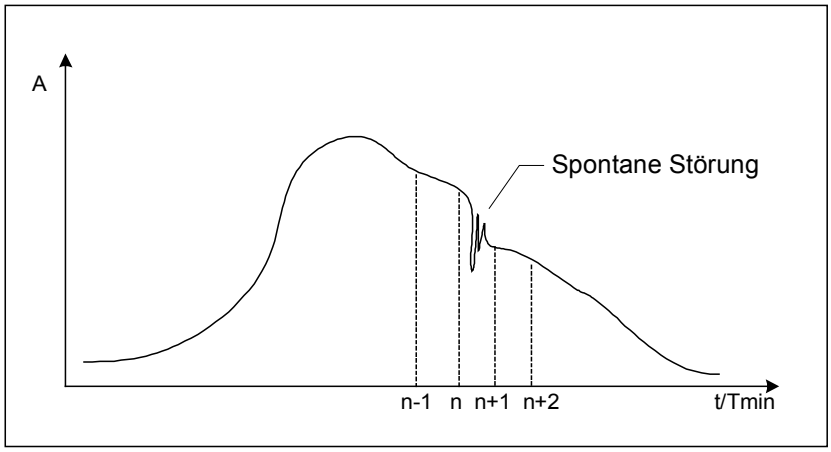


Abbildung 7.7: Ignorierung der spontanen Störung

Wie in der Abbildung 7.7 gezeigt, wird das Signal in dem minimalen Zeitabstand T_{min} abgetastet. Die Störung, die zwischen n-ter und (n+1)-ter Abtastung auftaucht, wird nicht erfaßt. Mit Einführung einer minimalen Zykluszeit wird somit der Einfluß einer spontanen Änderung ignoriert oder erheblich reduziert.

Die Einstellung der Abtastzeit muß bei einer praktischen Anwendung sehr genau analysiert werden. Bei einer zu langen Abtastzeit wird die Information der Strecke nicht genügend oft übertragen. Das bekannte Aliasing-Problem kann hier dann auftreten. In mancher Anwendung, z.B. bei einer Regelung, kann es im ernsten Fall zur Instabilität führen.

Außer Zähler und minimaler Zykluszeit gibt es 4 Bytes, die für den alten Prozeßwert, nämlich für den Prozeßwert bei der letzten Abfrage, reserviert sind. Durch einen binären Vergleich wird hier eine Änderung des Prozeßwertes festgestellt. Dabei wird ein Senden bei unverändertem Prozeßwert an das Empfänger-Automatisierungssystem weiter verzögert.

Ebenfalls werden auch Speicherzellen für die Adresseninformationen aus dem Sender-DB reserviert. Mit dessen Hilfe weiß das Sender-Automatisierungssystem, woher die zu sendende Information zu holen ist.

Entscheidungskriterien

Die Entscheidungskriterien, ob ein aktueller Prozeßwert gesendet werden soll oder nicht, hängen in diesem Modell von 2 Faktoren ab:

- einer Änderung des Prozeßwertes,
- einer Überschreitung der minimalen Wartezeit.

Dieser Entscheidungsvorgang kann durch das in der Abbildung 7.8 gezeigte Flußdiagramm verdeutlicht werden. In jedem Fetchauftrag wird zuerst die Zeit bewertet, ob die minimale Zeitperiode schon überschritten ist. Falls ja, wird anschließend eine Wertkontrolle unternommen. Erst nach der Feststellung der Wertänderung wird der Sendeauftrag eingeleitet.

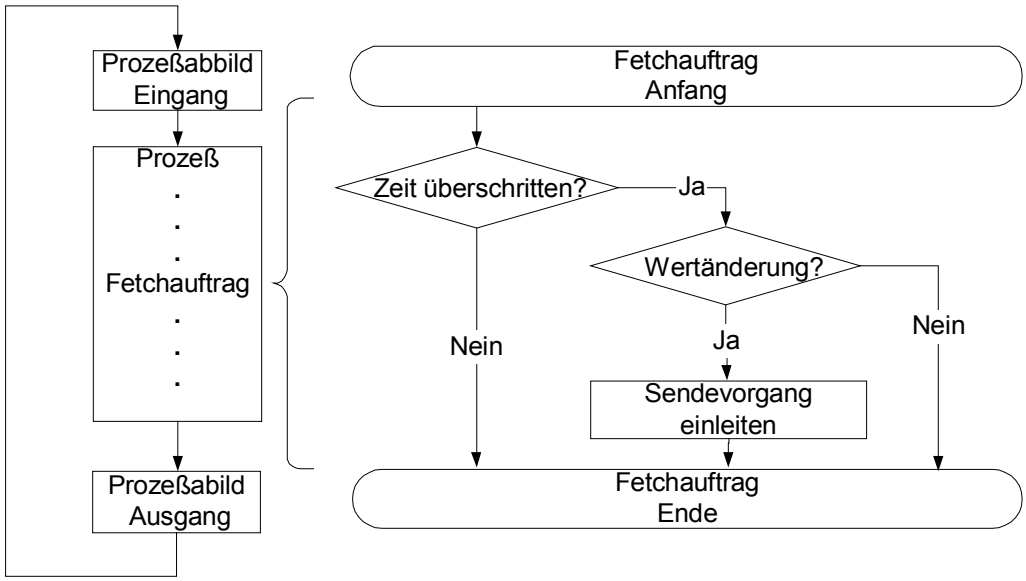


Abbildung 7.8: Entscheidungslogik im Fetch-Mechanismus

Es sind auch andere Kriterien denkbar. Z.B. wird durch die Einführung eines Quality Byte das Automatisierungssystem in die Lage gesetzt, die Qualität der Information zu beurteilen. Daraus kann entschieden werden, ob die Information noch sinnvoll ist.

Blockung

Der Fetchauftrag wird in einem bestimmten OB aufgerufen. Falls es viele Einträge im Fetchstack gibt, wie es in einem großen Projekt durchaus der Fall sein kann, wird viel Zeit für den Fetchauftrag in Anspruch genommen. Die hat den Nebeneffekt, daß andere Anwendungsbausteine im gleichem OB möglicherweise nicht oder zu wenig berücksichtigt werden.

Um diesen potentiellen Engpaß zu beseitigen, wird eine "Blockung" durchgeführt. Dabei wird eine vorher definierte Anzahl von Eintragungen in einem Zyklus überprüft. Bei den nachfolgenden Zyklen werden die restlichen Einträge nach der Reihenfolge weiterverarbeitet (Abbildung 7.9).

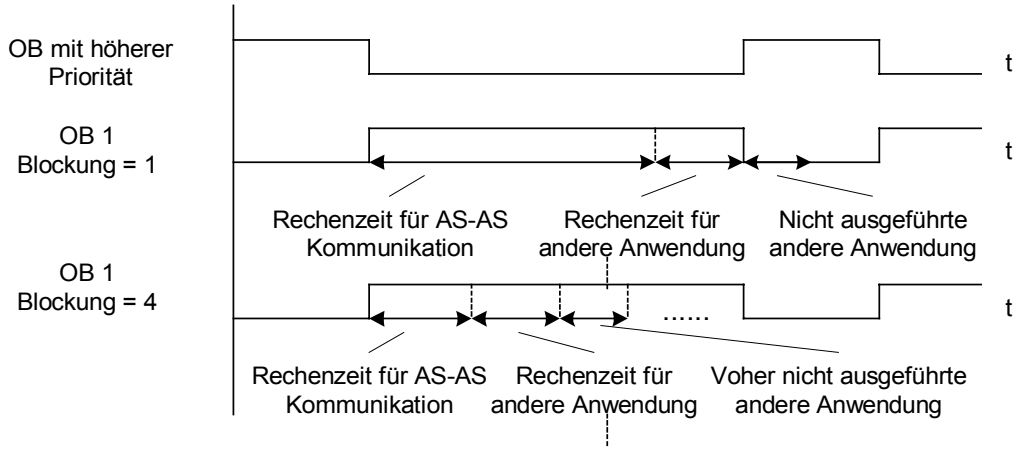


Abbildung 7.9: Einfluß der Blockung

Durch diese Methode bleibt die Rechenzeit für die Verarbeitung eines Fetchauftrages kalkulierbar, weil die Anzahl von auf einmal verarbeiteten Einträgen begrenzt ist. Somit kann es sichergestellt werden, daß in jedem Zyklus ein Teil von dem gesamten Zyklus für andere Anwendungen zur Verfügung steht.

7.3 Empfangsseite

7.3.1 Architektur beim Empfänger

Auf dem Empfänger-Automatisierungssystem wird die RECEIVE-FC zyklisch aufgerufen. In der RECEVE-FC selbst wird der Kommunikationsbaustein BRCV intern aufgerufen. Es wird somit ständig kontrolliert, ob irgendwelche Daten angekommen sind. Falls ja, wird das vom Sender-Automatisierungssystem gesendete Telegramm in einen Datenpuffer geschrieben. Die Unterscheidung zu möglichen anderen existierenden Sender-Automatisierungssystemen geschieht durch die interne Vergabe von unterschiedlichen IDs.

Ein Command-Interpreter entpackt das empfangene Telegramm. Durch die im Telegramm enthaltene Adresseninformation wird das entsprechende Ziel lokalisiert, das normalerweise ein Datenbaustein (DB) ist. Ist der DB vorhanden, schreibt der Command-Interpreter die empfangenen Daten in den DB. Dabei wird im Automatisierungssystem sichergestellt, daß der Eingangspuffer für ein Telegramm nicht überschrieben wird, bevor das Telegramm komplett abgearbeitet werden konnte. In dem Sender-Automatisierungssystem muß ein daraus eventuell erfolgtes Abweisen des Telegramms erkannt und das Telegramm wiederholt gesendet werden. Auf diese Weise wird ein Telegrammverlust verhindert (auch bei schnell aufeinander folgenden Sendungen).

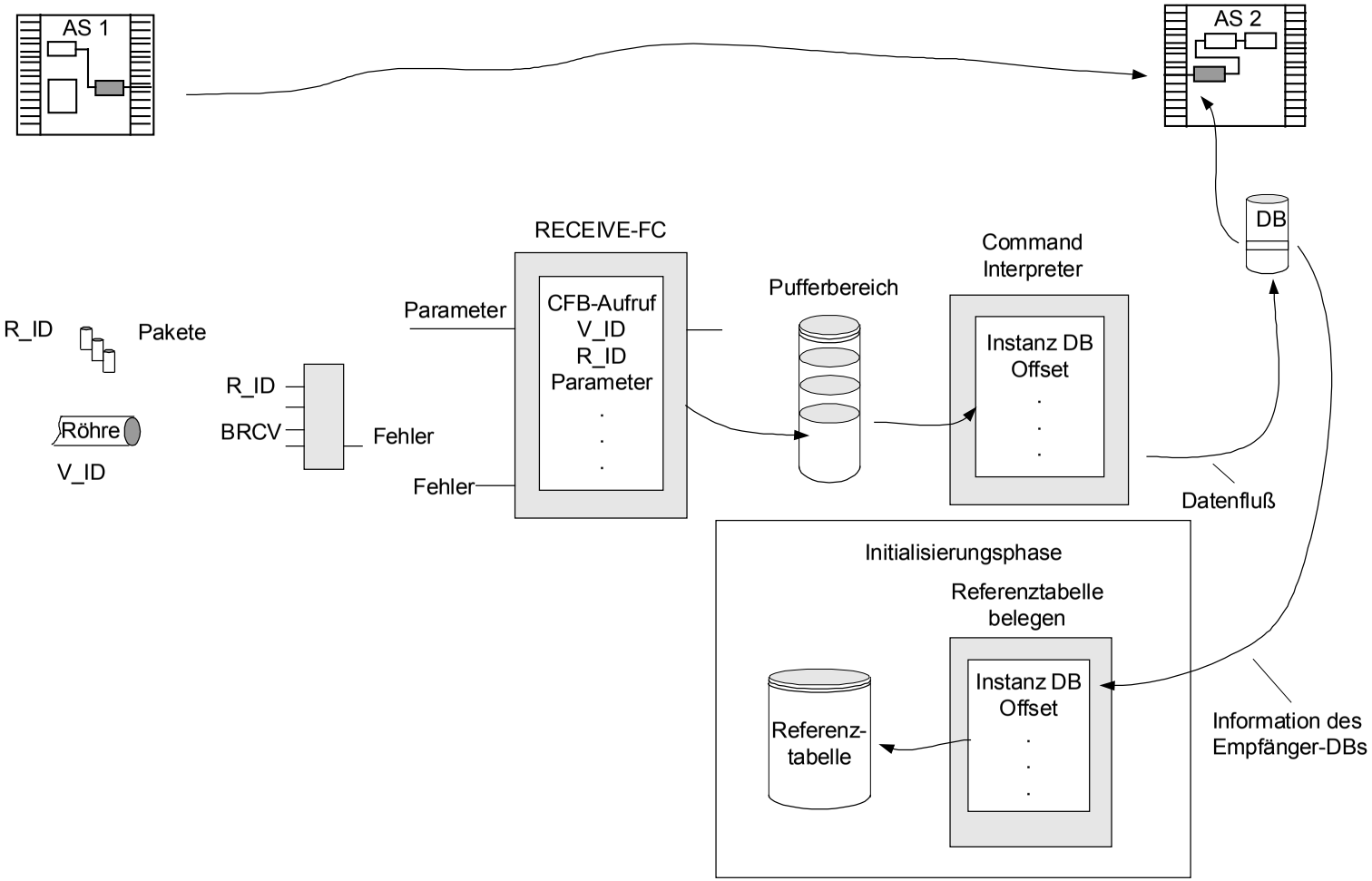


Abbildung 7.10: Kommunikationsablauf in einem Empfänger-Automatisierungssystem

Zu jedem Sender-Automatisierungssystem existiert auf dem Empfänger-Automatisierungssystem eine zugehörige Referenztabelle. In dieser Tabelle stehen sämtliche Informationen, mit deren alle empfangswillige DBs auf Empfänger-Automatisierungssystem identifiziert werden. Die Informationen über den Empfänger-DB bekommt man mit Hilfe der projektierten Verbindungen. Sie werden dann durch eine FC oder einen übergeordneten Compiler in der Initialisierungsphase in eine Referenztabelle geschrieben.

Eine Referenztabelle ist deswegen nötig, weil eine Lokalisierung der Information angestrebt wird. Der Inhalt wird vor der ersten zyklischen Datenübertragung durch ein Sondertelegramm zum Sender-Automatisierungssystem gesendet, wodurch eine sendeseitige Telegrammbildung erst möglich wird.

7.3.2 Belegung der Referenztabelle

In der Referenztabelle wird die Information über das Übertragungsziel und die Übertragungseinstellung gespeichert. Mit der Übertragungseinstellung ist in diesem Fall der minimale Übertragungszyklus gemeint, um den das Sender-Automatisierungssystem vor erneutem Senden unbedingt warten muß. Wie im Fetchstack wird die Referenztabelle auch in einer bestimmten Reihenfolge belegt (Abbildung 7.11):

Am Anfang steht ein Byte für die Summe der tatsächlich belegten Einträge. Danach folgt der Inhalt der Einträge.

0	Summe der Einträge	reserviert
2	Destination DB	
4	Destination Offset	
6	Destination Länge	
8	reserviert	Minimaler Zyklus
10	reserviert	reserviert
12	(Source DB)	
14	(Source Offset)	
16	Zweiter Eintrag ..	
30	Dritter Eintrag ..	

Abbildung 7.11: Belegung der Referenztabelle

Da die Zielinformation sich um die Bausteine auf Empfänger-Automatisierungssystem handelt, wird sie lokal auf das Empfänger-Automatisierungssystem gelegt. Sie wird vom übergeordneten Compiler in dem Ladevorgang oder von einer FC in der Initialisierungsphase erzeugt und belegt die ersten 6 Bytes eines Eintrags, der insgesamt 14 Bytes lang ist. Danach gibt es ein Byte für den minimalen Zyklus. Diese Zielinformation wird nach dem Verbindungsaufbau durch Sondertelegramme an das Sender-Automatisierungssystem übertragen.

In der Referenztabelle sind noch Speicherzellen für die Sendeinformation reserviert, die die Herkunft der Daten dokumentiert. Das ist vor allem als eine Vereinfachung bei der Änderungsprojektierung gedacht, die später noch genauer erläutert wird.

7.4 Allgemeine Speicherbelegung

Um das IK Modell zum Laufen zu bringen, werden unterschiedliche System-DBs eingeführt, die beim Setzen eines Startbits oder bei einem CPU-Zustandswechsel von NEUSTART zu RUN in der Anlaufphase automatisch generiert werden. Die wichtigsten werden in der Abbildung 7.12 dargestellt und im folgenden beschrieben:

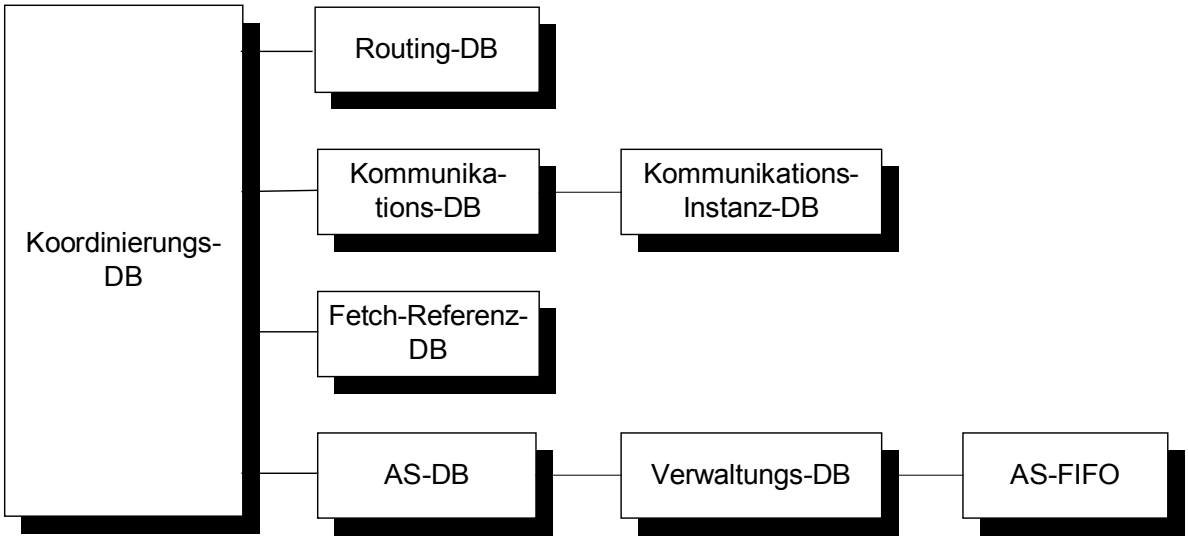


Abbildung 7.12: Notwendige Speicherbelegung im IK Modell

Koordinierungs-DB

Der Koordinerungs-DB enthält u.a. Speicherplätze für administrative Variable. Der Datenbaustein verwaltet die zentrale Einstellung der Kommunikation zwischen den Automatisierungssystemen.

Routing-DB

Der Routing-DB speichert den aktuellen Kommunikationszustand zu allen Automatisierungssystemen sowie zu allen lokalen Verbindungen.

Kommunikations-DB

Der Kommunikations-DB dient zum Zwischenspeichern des Telegramms (zu sendende und empfangende) und beinhaltet die Kontrollbits zur Überprüfung der Übertragung je nach der Verbindung.

Kommunikationsinstanz-DB

Der Kommunikationsinstanz-DB bietet einen verbindungs-granularen Pufferbereich für die Versendung und den Empfang der Telegramme.

Fetch-Referenz-DB

Der Fetch-Referenz-DB dient zum Speichern der Information der sendewilligen und empfangswilligen Bausteine sowie deren Übertragungskriterien.

AS-DB

Der AS-DB speichert die administrative Information über alle Automatisierungssysteme.

Verwaltungs-DB

Der Verwaltungs-DB gibt an, welcher Speicherbereich einem Automatisierungssystem zugeteilt ist.

AS-FIFO

Pro Verbindung zu einem Kommunikationspartner wird im IK Modell ein verbindungs-granularer Datenbaustein für zwei priorisierte FIFOs unterschiedlicher Länge und einige spezifische Steuerinformationen eingerichtet. Die Länge der Sendepuffer ist abhängig von der Art der Übertragung im Automatisierungsgerät.

7.5 Arbeitsvorgang

Je nach Aufgabenbereich kann der Kommunikationsvorgang zwischen den AS in 3 Phasen gegliedert werden, Anlauf-, Koordinierungs- und Betriebsphase.

Anlaufphase

Der Ladevorgang für ein Sender- und ein Empfänger-Automatisierungssystem läuft jeweils asynchron. Es gibt immer ein Zeitversatz zwischen zwei Ladevorgängen. Wegen dieses Zeitversatzes muß eine Methode gefunden werden, daß die vorher sich im Netz befindenden AS automatisch das neu ins Netz hinzugefügte AS entdecken können. Nach der Entdeckung erfolgt dann eine entsprechende Initialisierung in allen Automatisierungssystemen im Netz in Hinblick auf eine künftig mögliche Kommunikation mit diesem Automatisierungssystem.

Koordinierungsphase

Die Information über die projektierte Kommunikationsbeziehung ist nach dem Laden jeweils auf ein Sender- und ein Empfänger-Automatisierungssystem verteilt. Um alle Information zur Bildung eines Telegramms zwischen den Kommunikationspartnern zu bekommen, muß das Sender-Automatisierungssystem auch über die Kommunikationsinformation auf der Empfängerseite verfügen. Deswegen ist ein Informationsabgleich notwendig, bevor der Prozeßwert überhaupt übertragen werden kann.

Betriebsphase

Nachdem das Fetchstack mit der Kommunikationsinformation gefüllt wird, kann die Datenübertragung für Prozeßwerte beginnen. Es geht in die normale Betriebsphase.

7.5.1 Anlaufphase

Nach der Generierung und dem Laden der Codes kann das Automatisierungssystem in Betrieb genommen werden. Bevor eine logische Kommunikationsverbindung zustande kommt, gibt es eine Synchronisationsphase zwischen beiden benachbarten Kommunikationsteilnehmern.

In bestimmten Zeitabständen (z.B. 30 s) sendet ein Automatisierungssystem ein Lebenszeichen, um allen anderen Nachbar-Automatisierungssystemen mitzuteilen, daß er erreichbar ist. Wenn ein Nachbar-Automatisierungssystem das Lebenszeichen empfängt, sendet es ein Acknowledge-Telegramm zurück. Wenn das Acknowledge-Telegramm bei dem ursprünglichen Sender-Automatisierungssystem ankommt, wird eine logische Verbindung zwischen den beiden Automatisierungssystemen eingerichtet.

Jedes Automatisierungssystem bekommt bei der Projektierung eine Knotennummer zugeteilt. Diese Knotennummer entscheidet, welches Automatisierungssystem ein aktiver Busteilnehmer ist. Ein aktiver Busteilnehmer soll die Initialisierung zuerst einleiten, um einen möglichen Konflikt zu vermeiden. Hierbei wird ein Automatisierungssystem mit höherer Knotennummer als aktiver Teilnehmer vorausgesetzt.

Die Abbildung 7.13 zeigt den gesamten Ablauf beim Verbindungsaufbau zwischen dem Automatisierungssystem x und dem Automatisierungssystem y (x > y),

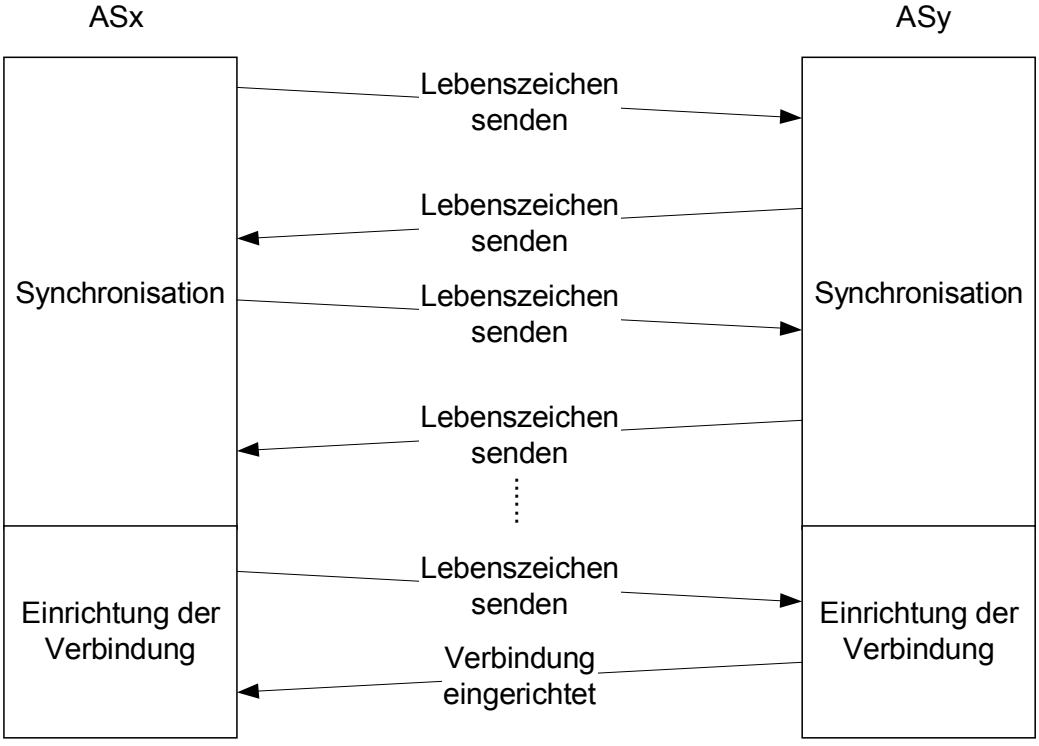


Abbildung 7.13: Verbindungsaufbau in der Anlaufphase

Die Kommunikation läuft mit BSEND/BRCV auf einer gesicherten, verbindungsorientierten Verbindung. Um eine zusätzliche Sicherheit zu erreichen, wird eine Telegrammnummer in jedem danach kommenden Telegramm hinzugefügt. Durch Kontrolle der Telegrammnummer kann das Empfänger-Automatisierungssystem wissen, ob das erwartete Telegramm angekommen oder verlorengegangen ist. Die dafür notwendige Initialisierung wird nach dem Verbindungsaufbau vorgenommen.

7.5.2 Koordinierungsphase

Nachdem eine Verbindung zwischen beiden Automatisierungssystemen etabliert ist, muß das Fetchstack mit der Zielinformation gefüllt werden, um eine Telegrammbildung für die Übertragung von Prozeßwerten zu ermöglichen. Das geschieht durch Austausch von einem Sondertelegramm in der Koordinierungsphase direkt nach dem Verbindungsaufbau.

Das aktive Automatisierungssystem x sendet zuerst ein Referenz-Telegramm zu seinem Partner-Automatisierungssystem y. Im Referenz-Telegramm ist die Information über die aktuelle Belegung des für Automatisierungssystem y zugeordneten Fetchstacks vorhanden. Es gibt hier 3 Sorten von Einträgen:

- schon vorhandene Verbindung in Fetchstack,
- gelöschte Verbindung in Fetchstack,
- neue Verbindung in Fetchstack.

Anhand dieser Information wird die zugehörige Referenztabelle auf dem Partner-Automatisierungssystem y aktualisiert. Danach sendet das Partner-Automatisierungssystem y ein Source-Telegramm und ein Destination-Telegramm zum Automatisierungssystem x. In den beiden Telegrammen stehen folgende Informationen:

- schon vorhandene Verbindung in Referenztabelle,
- gelöschte Verbindung in Referenztabelle.
- neue Verbindung in Referenztabelle,

Das aktive Automatisierungssystem x empfängt die beiden Sondertelegramme und reorganisiert das dem Automatisierungssystem y zugeordnete Fetchstack erneut.

Weil jedes Automatisierungssystem gleichzeitig Sender oder Empfänger sein kann, wird der gleiche Vorgang auch in der anderen Richtung wiederholt: Es werden Fetchstack auf dem Automatisierungssystem y und die Referenztabelle auf dem Automatisierungssystem x auch neu organisiert. Diesen Vorgang zeigt die Abbildung 7.14.

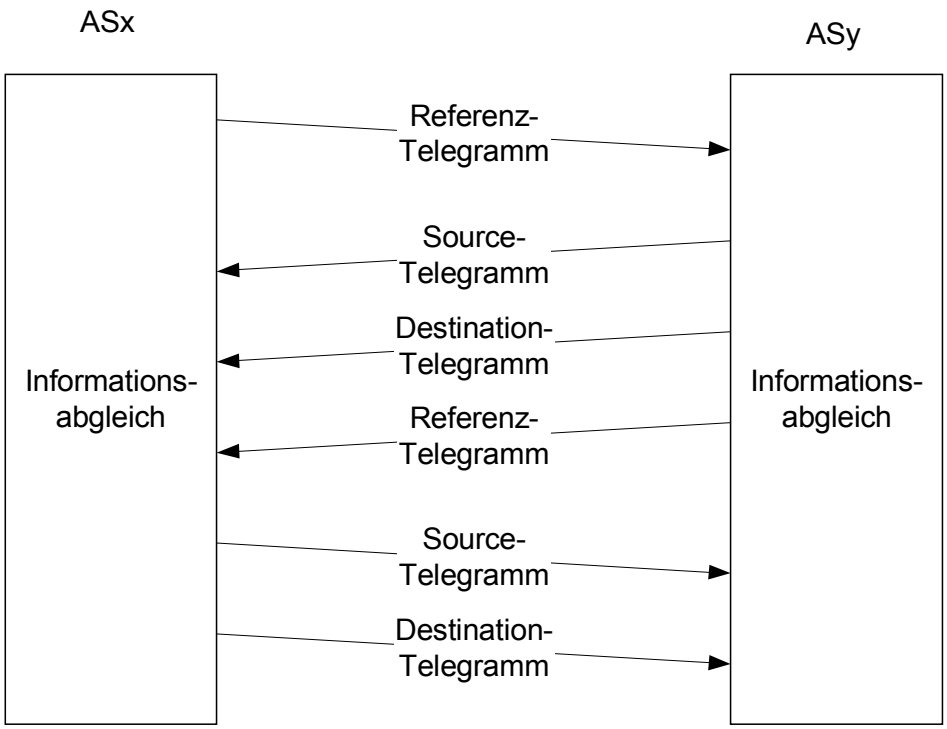


Abbildung 7.14: Informationsaustausch in der Koordinierungsphase

Nach diesem Informationsaustausch besitzen die beiden Automatisierungssysteme alle Informationen, die für eine erfolgreiche Datenübertragung notwendig sind. Dann folgt die Betriebsphase, in der der Datenaustausch für Prozeßwerte angestoßen werden kann.

7.5.3 Betriebsphase

Die Aufgabe in der Betriebsphase besteht darin, nach bestimmten Kriterien die Prozeßwerte, deren Anschlüsse eine geräteübergreifende Verschaltung besitzen, zwischen den Kommunikationspartnern zu übertragen.

Das Fetchstack wird zyklisch kontrolliert. Bei Erfüllung der Sendebedingung wird der gewünschte Prozeßwert in ein Standardtelegramm gepackt und über die Leitung zum Empfänger-Automatisierungssystem geschickt. Der Sendezyklus für einen Prozeßwert hängt einerseits von der vorgeschriebenen minimalen Zykluszeit ab. Andererseits muß eine Wertänderung vorliegen.

Falls das Sender-Automatisierungssystem eine geräteübergreifende Verbindung zu mehreren Automatisierungssystemen hat (Abbildung 7.15), werden die Partner-Automatisierungssysteme nacheinander kontrolliert, ob eine Sendeanforderung vorliegt. Die Einführung von Blockung hat hier auch eine praktische Bedeutung. Es wird somit vermieden, daß ein zuerst kontrolliertes AS die Übertragung von Prozeßwert nach einem später kontrollierten AS blockiert.

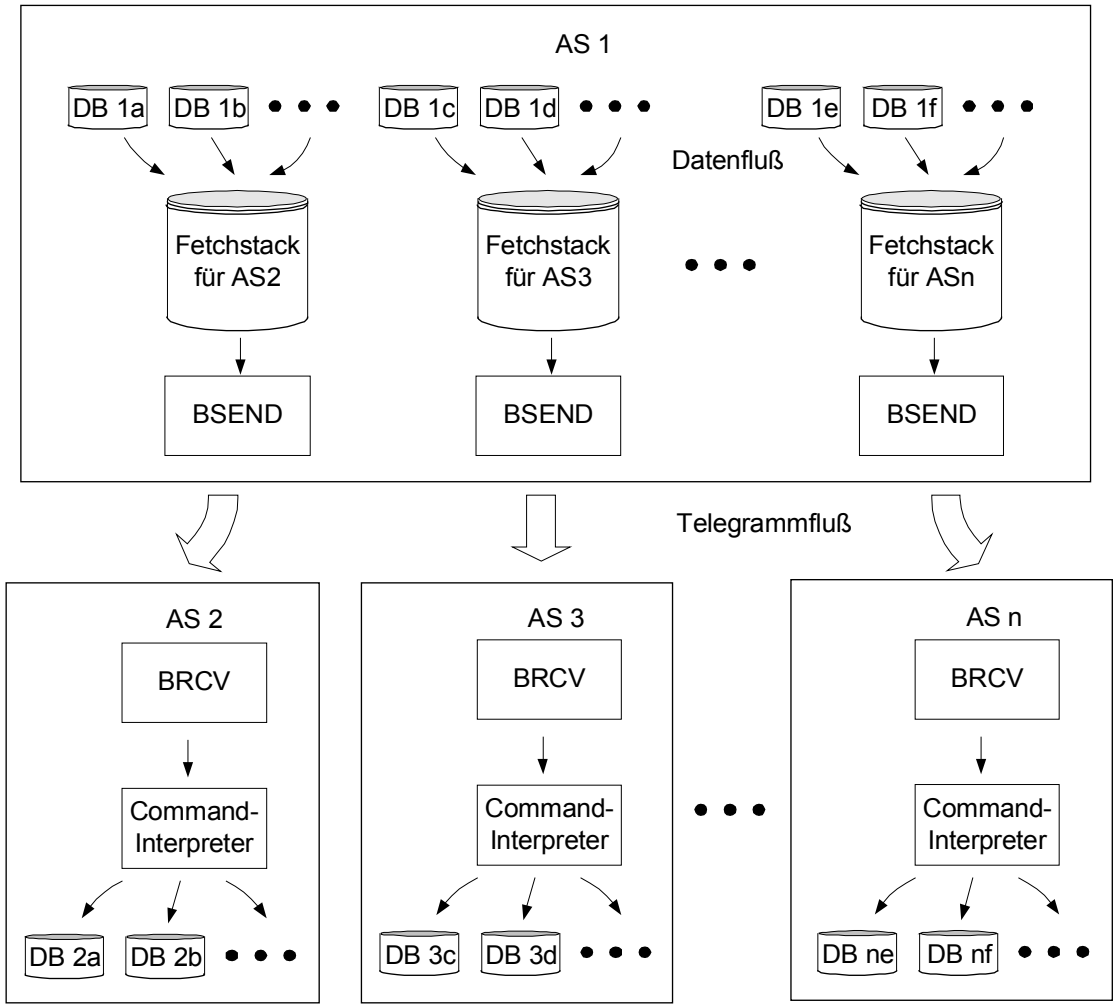


Abbildung 7.15: Datenaustausch in der Betriebsphase

7.6 Änderungsprojektierung

Für eine Verbindung gibt es zwei Kommunikationspartner. Falls der Zustand der Verbindung sich geändert hat, handelt es sich dann um eine Modifizierung des Informationsstandes bei beiden AS. Die Konzepte in Kapitel 5 und Kapitel 6 sehen vor, die vorgenommene Änderung sowohl auf dem Sender-AS als auch auf dem Empfänger-AS zu laden. Falls viele Änderungen, die die Kommunikation mit anderen AS betreffen, auf dem Sender-AS vorgenommen werden sollen, bedeutet dies eine Vielzahl von Ladevorgängen (Abbildung 7.16), die bezüglich der in Kapitel 4 genannten Kriterien unerwünscht sind. Hier wird das Ziel angestrebt, die Anzahl möglicher Ladevorgänge zu reduzieren.

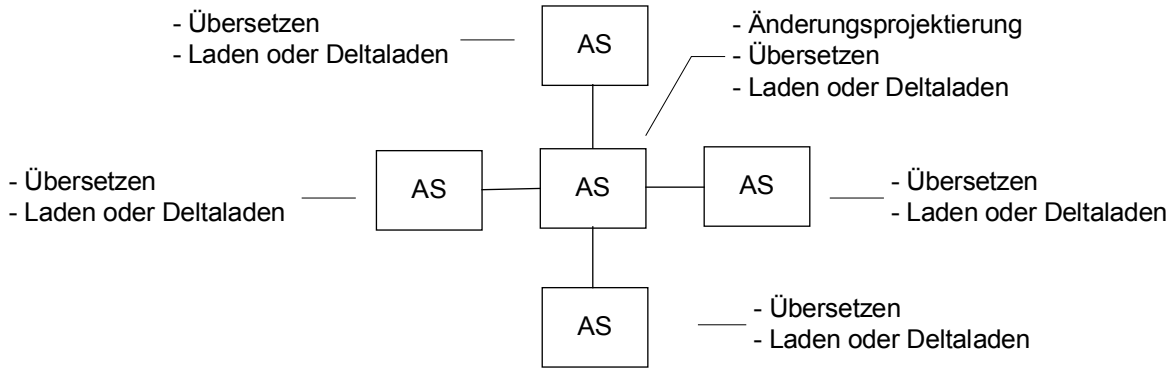


Abbildung 7.16: Konventionelle Methode bei der Änderungsprojektierung

Bei dieser Untersuchung wird zwischen Änderung der Kommunikation auf der Sendeseite und auf der Empfangsseite unterschieden.

7.6.1 Änderung auf der Sendeseite

Falls eine Änderung beim sendewilligem Baustein vorgenommen wird, wird ein übergeordneter Compiler das erkennen und durch Laden oder Deltaladen die betroffenen Einträge im Fetchstack modifizieren oder neu einfügen. Wie bei der erstmaligen Inbetriebnahme wird der Ladeinhalt unterschieden nach 3 Typen von Einträgen:

- Alter Eintrag entspricht den Verbindungen ohne Änderung.
- Gelöschter Eintrag entspricht den gelöschten Verbindungen.
- Neuer Eintrag entspricht den neu hinzugefügten Verbindungen.

Im Betrieb kontrolliert ein Automatisierungssystem ständig, ob eine Änderung im Fetchstack auftaucht, die in Folge einer Projektierungsänderung der Verbindung verursacht wurde. Falls ja, wird die Änderung in einem Sondertelegramm (Delta-Fetch-Telegramm) erfaßt und zum Partner-Automatisierungssystem verschickt. Die Abbildung 7.17 zeigt diesen Ablauf.

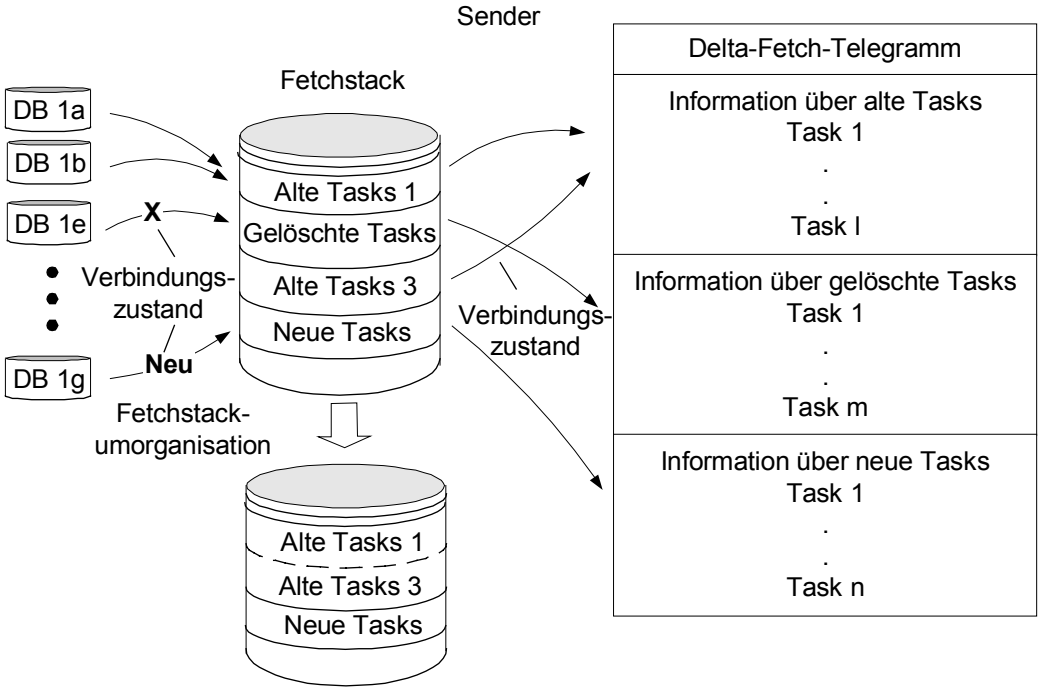


Abbildung 7.17: Ablauf einer Änderungsprojektierung auf der Sendeseite

Beim Löschen einer Verbindung wird im Fetchstack eine unbenutzte Speicherlücke entstehen. Falls in einem großen Projekt sehr viel umprojektiert wird, kann es wegen dieser Speicherlücken leicht zu Engpässen kommen. Um ein Fetchstack sinnvoll zu benutzen, wird nach der Verschickung eines Sondertelegramms eine Kontrolle durchgeführt. Falls unbenutzte Speicherlücken entdeckt werden, generiert das Automatisierungssystem das Fetchstack sofort neu und gibt den unbenutzten Speicherbereich frei. Das sieht man auch in der Abbildung 7.17.

Das Empfänger-Automatisierungssystem empfängt das Delta-Fetch-Telegramm. Der Command-Interpreter entpackt es und ruft eine FC auf, die anhand der im Sondertelegramm stehenden Information eine neue Generierung der zugehörigen Referenztabelle einleitet. Die Information einer gelöschten Verbindung wird aus der Referenztabelle entfernt. Die Information einer neuen Verbindung wird am Tabellenende hinzugefügt. Der Vorgang wird in der Abbildung 7.18 dargestellt.

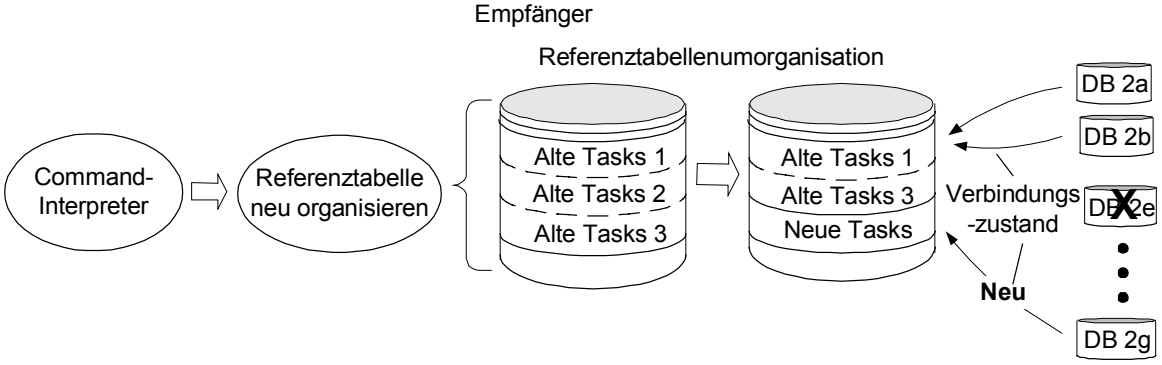


Abbildung 7.18: Ablauf einer Änderungsprojektierung auf der Empfangsseite

7.6.2 Änderung auf der Empfangsseite

Falls eine Änderung auf der Empfangsseite auftaucht, d.h. der Zustand des Verbindungsanschlusses im Empfangsbaustein geändert wird, soll auch die Referenztabelle aktualisiert werden. Das geschieht wie bei der Änderung auf der Sendeseite durch einen übergeordneten Compiler während des Ladevorgangs.

In jedem OB1-Zyklus wird kontrolliert, ob die Einträge in der Referenztabelle sich geändert haben. Die mögliche Änderung muß auch dem Fetchstack im Sender-Automatisierungssystem mitgeteilt werden, damit das Sender-Automatisierungssystem in künftigen Zyklen die gewünschten Prozeßwerte senden kann. Analog zur Änderung auf der Sendeseite wird hier ein Delta-Referenz-Telegramm eingeführt. In diesem Sondertelegramm steht die Information, welche Änderung in der Referenztabelle existiert (Abbildung 7.19). Nach der Sendung des Sondertelegramms wird die Referenztabelle auch neu organisiert, um den nicht mehr benutzten Speicherbereich freizugeben.

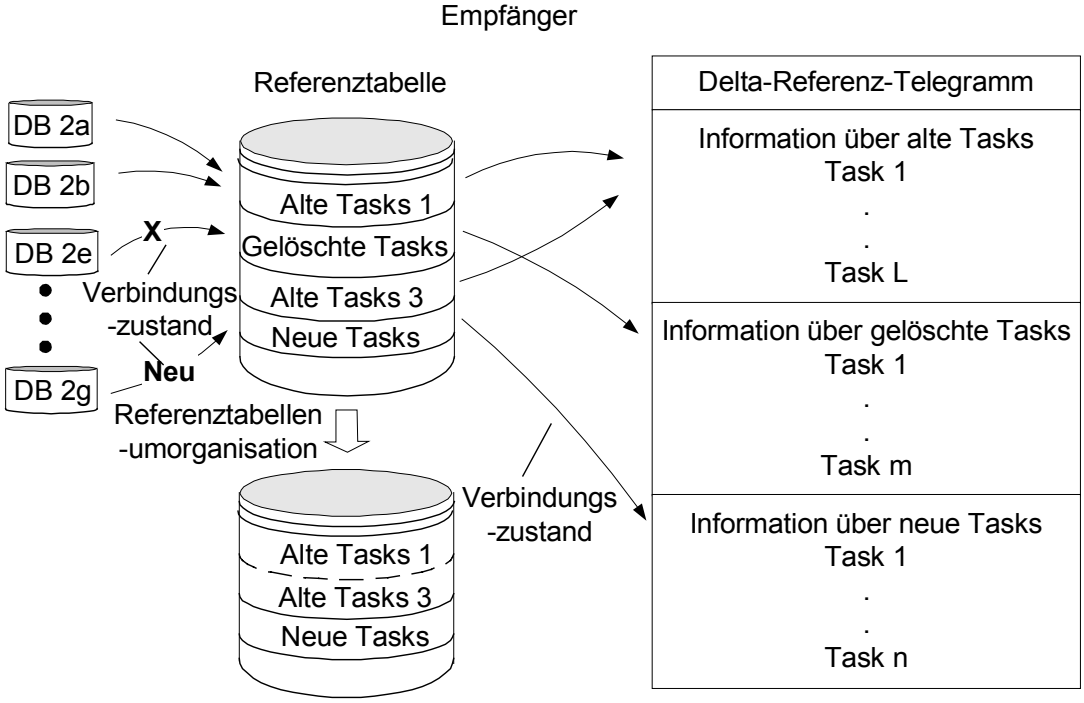


Abbildung 7.19: Änderung einer Änderungsprojektierung auf der Empfangsseite

Das Sender-Automatisierungssystem empfängt das Delta-Referenz-Telegramm und ruft eine FC auf, um das Fetchstack mit Hilfe der empfangenen Informationen zu reorganisieren. Gelöschte Einträge werden entfernt, um in Anspruch genommenen Speicherplatz in Fetchstack freizugeben. Neue Einträge werden am Ende des Fetchstacks hinzugefügt (Abbildung 7.20).

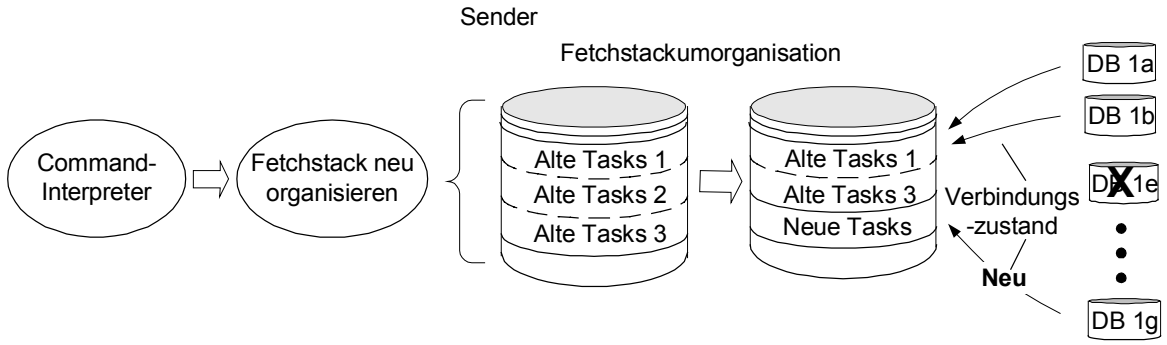


Abbildung 7.20: Umorganisation des Fetchstacks

In der Abbildung 7.21 wird der Austausch von Sondertelegrammen gezeigt, die eine Änderung sowohl auf der Sendeseite als auch auf der Empfangsseite initiieren.

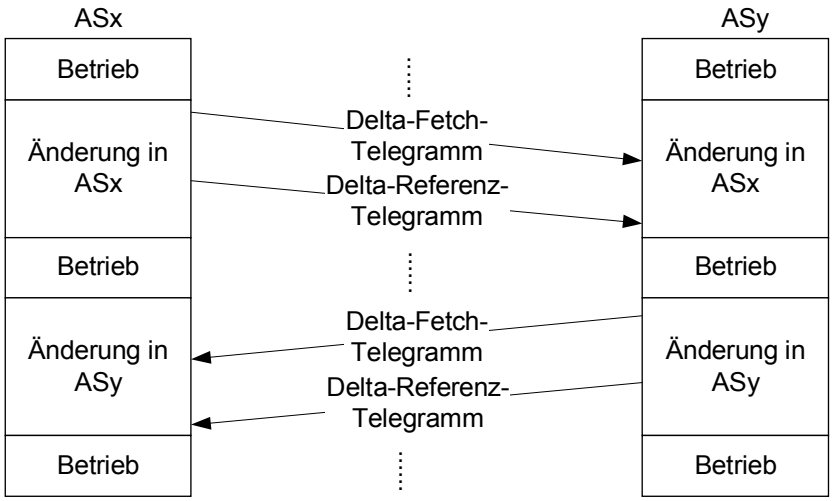


Abbildung 7.21: Informationsaustausch bei Änderungsprojektierung

Diese Vorgehensweise eignet sich besonders gut für die Änderungsprojektierung. Die neue Übersetzung und das Deltaladen ist somit nur dort notwendig, wo die Projektierungsänderung vorgenommen wird. Die Projektierungsänderung, deren Information auch andere Automatisierungssysteme betreffen, z.B. Empfangsadresse, wird dem Partner-Automatisierungssystem ohne weiteres im Betrieb durch ein Sondertelegramm mitgeteilt. Der Kommunikationspartner wird im Zustand RUN seinen Informationsstand im Fetchstack oder in der Referenztabelle aktualisieren. Dort ist keine nochmalige Übersetzung oder ein Deltaladen notwendig (Abbildung 7.22).

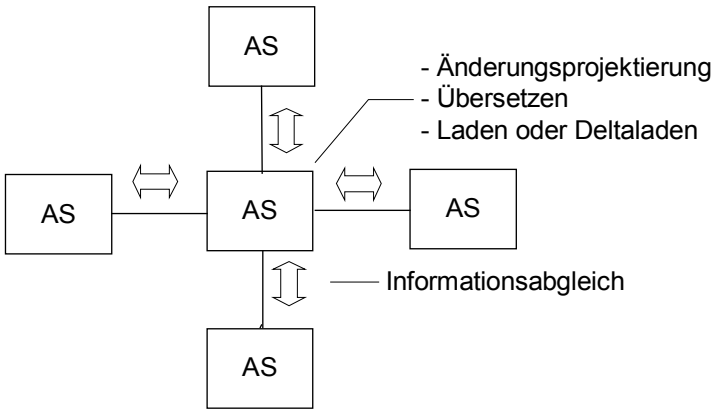


Abbildung 7.22: Änderungsprojektierung bei IK

7.7 Einige Besonderheiten des IK Modells

Im IK Modell ist es möglich, die im Modell mit blockorientierter Übertragung eingeführte Methode bezüglich Überwachung, Ersatzwert und Fehlerreaktion einzusetzen. Aus dieser Eigenschaft des IK Modells ergeben sich neue Möglichkeiten, z.B. hier läuft der Datenaustausch in Form von Telegrammen in einem standardisierten Format ab. Durch Einpacken von zusätzlichen Informationen oder Einführung von Sondertelegrammen hat man in folgenden Bereichen eine bessere Performance:

- eine bessere Kontrolle des Kommunikationsablauf,
- eine effektivere Verwaltung der Kommunikationsressourcen,
- eine Erhöhung der Übertragungssicherheit.

7.7.1 Routing

Bevor die Daten zum Partner-Automatisierungssystem geschickt werden, soll die Verbindung noch projektiert werden. Die Verbindung kann nur dazu genutzt werden, die Daten zum vorher festgelegten Automatisierungssystem zu transportieren. Zwischen zwei AS können mehrere Verbindungen projektiert werden. Die Eindeutigkeit wird durch die Einführung von V_ID garantiert. Durch die Verbindungsprojektierung werden die Ressourcen für die jeweilige Verbindung reserviert.

Der Kommunikationspartner kann ein benachbartes oder ein entferntes Automatisierungssystem sein. Für eine Verbindung zwischen entfernten Automatisierungssystemen müssen auf jedem dazwischen liegenden Automatisierungssystem die Ressourcen auch alle reserviert werden. Als Beispiel werden in der Abbildung 7.23 vier Automatisierungssysteme dargestellt. Falls AS 1 mit einem anderen Automatisierungssystem kommunizieren möchte, werden sowohl auf AS 1 als auch auf AS 2

3 mal Ressourcen belegt, die jeweils für die Kommunikation zwischen AS 1 und AS 2, AS 1 und AS 3, AS 1 und AS 4 zum Einsatz kommen.

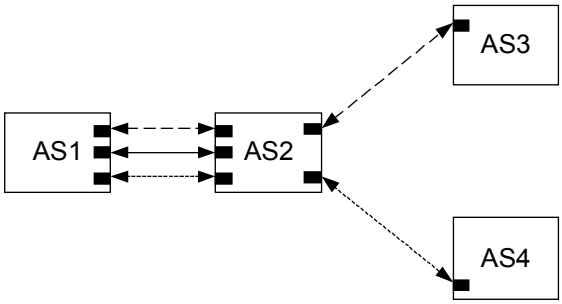


Abbildung 7.23: Automatisierungssystem mit projektierter Verbindung

Die Anzahl der Verbindungsressourcen auf einem Automatisierungssystem ist beschränkt und hängt von der eingesetzten CPU ab. Damit verbunden ist auch eine Begrenzung der maximal projektierbaren Verbindungen. Die Einschränkung ist besonders ungünstig für sternförmige Netze. Falls z.B. in der Abbildung 7.24 jedes Automatisierungssystem mit dem anderen Automatisierungssystemen kommunizieren möchte, werden Ressourcen auf dem in der Mitte liegenden Automatisierungssystem wegen dieses sternförmigen Netzwerks stark verbraucht. Im Ernstfall ist die Kapazität sogar erschöpft, wenn nicht genügend Verbindungen auf dem mittleren Automatisierungssystem projektiert werden können.

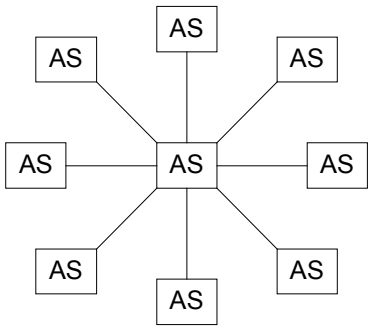


Abbildung 7.24: Engpaß der Ressourcen auf dem mittleren Automatisierungssystem

Um diesen Engpaß zu vermeiden, wird beim IK Modell ein besonderer Mechanismus für das Routing eingeführt. Jeder Datensatz innerhalb eines Telegramms wird vom Empfänger-Automatisierungssystem gelesen. Durch einen Entscheider wird der Datensatz zum angesprochenen DB weitergeleitet, falls das Empfänger-Automatisierungssystem selbst die Endstation ist. Wenn nicht, wird gesucht, zu welchem Nachbar-Automatisierungssystem der Datensatz weitergeleitet werden soll. In einem für dieses gefundene Nachbar-Automatisierungssystem vorgesehenen Übergangs-FIFO wird der Datensatz gespeichert. Im nächst möglichen Schritt werden die Daten aus dem Übergangs-FIFO, gegebenenfalls noch die vom aktuellen Automatisierungssystem für dieses Nachbar-Automatisierungssystem

generierten Daten, geholt und weiter übertragen. Damit ist jedes Automatisierungssystem in der Lage, die nicht für sich bestimmten Datensätze zu einer entsprechenden Zwischenstation weiter zu "routen", bis die Daten ans Ziel gekommen sind. Diese Methode wird in der Abbildung 7.25 gezeigt.

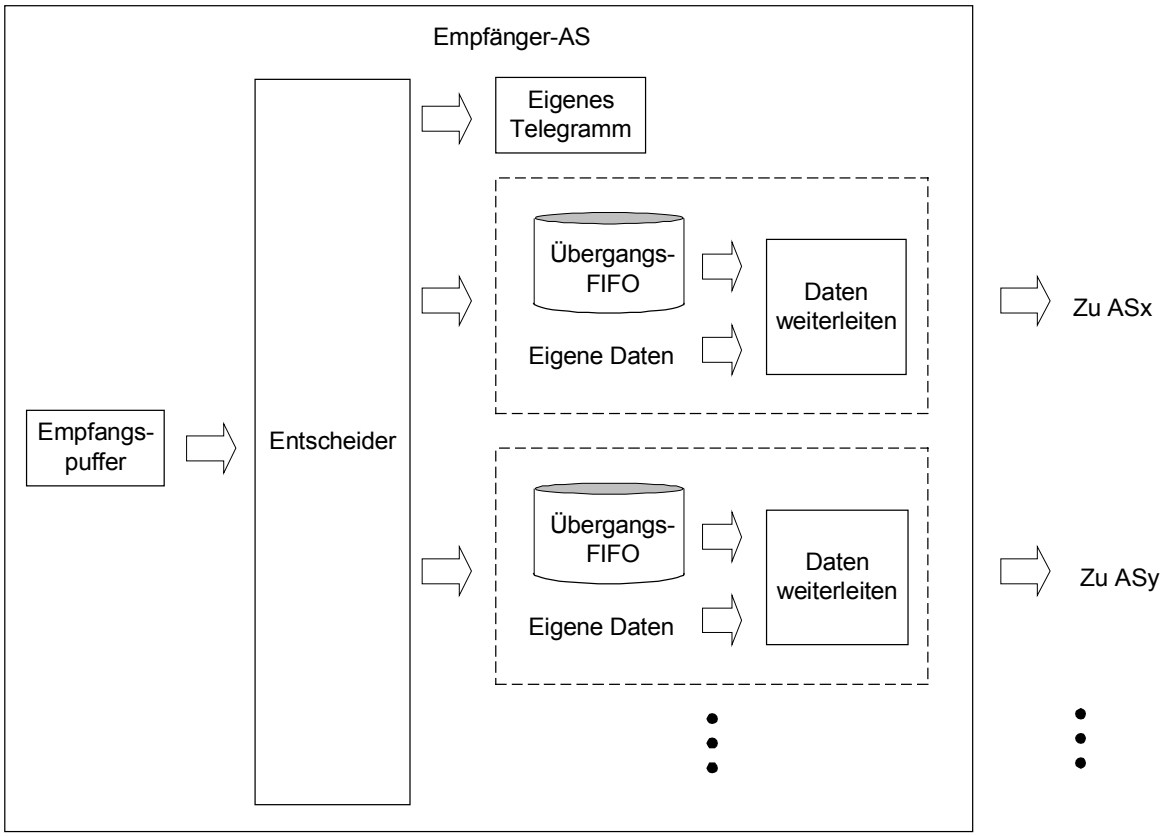


Abbildung 7.25: Routingmechanismus

In den ersten beiden Konzepten wird die Zieladresse in Form von Verbindungs-ID ausgedrückt. Hier wird sie ins Telegramm geschrieben und mitübertragen. Man braucht nur die Verbindung zwischen den benachbarten Automatisierungssystemen zu projektieren, um alle Automatisierungssysteme im Netz zu erreichen. Der daraus resultierende Vorteil ist der geringere Verbrauch von Verbindungsressourcen. Für die Verbindung zwischen zwei Kommunikationspartnern braucht man nicht mehr die Verbindungsressourcen auf allen dazwischen liegenden Automatisierungssystemen zu reservieren. Die Ressourcen sind nur für Nachbar-Automatisierungssysteme einzusetzen.

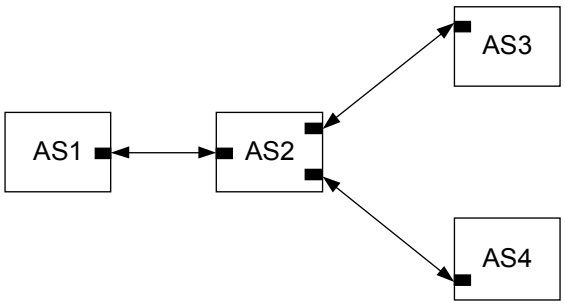


Abbildung 7.26: Routingfähiges Automatisierungssystem

7.7.2 Verbindungsüberwachung

Eine der wichtigsten Kriterien für ein Kommunikationskonzept im Bereich Automatisierungstechnik ist die Zuverlässigkeit des Netzes. Es kann nicht vermieden werden, daß eine Netzkomponente irgendwann einmal ausfällt. Es ist sehr wichtig, daß der Zustand der Verbindungen zu den Kommunikationsteilnehmern innerhalb des Netzes jederzeit transparent bleibt.

Durch Einführung eines neuen Überwachungsmechanismus besteht hier die Möglichkeit, die am Systembus angeschlossenen Automatisierungssysteme auf ihre Erreichbarkeit zu überwachen. Damit hat ein Automatisierungssystem stets einen aktuellen Überblick über den Zustand der Anlage.

Die Überwachungsfunktion wird von jedem Automatisierungssystem ausgeführt, wobei jedes Automatisierungssystem alle weiteren angeschlossenen Automatisierungssysteme überwacht, zu denen eine Kommunikationsverbindung besteht. Die Überwachung wird automatisch beim Anlauf des Automatisierungssystems aktiviert und erfolgt typischerweise in Zyklen zwischen 30 Sekunden und einer Minute.

Automatische Entdeckung einer unterbrochenen Verbindung

In der Praxis kommt es öfters vor, daß einige Automatisierungssysteme im Netz nicht mehr erreichbar sind. Die Ursache kann vielfältig sein. Das kann z.B bei einer lokalen Änderung der Netztopologie passieren. Oder es kann durchaus sein, daß das Automatisierungssystem nicht mehr funktionsfähig ist. Außerdem funktioniert der Bus zwischen den Automatisierungssystemen auch nicht immer fehlerfrei.

In diesem Fall, in dem ein Automatisierungssystem nicht mehr ansprechbar ist, sollen andere Automatisierungssysteme dann gestoppt werden, zum oder über das ausgefallene Automatisierungssystem Telegramme weiter zu senden. Diese Aktion verschwendet einerseits die Rechenzeit und andererseits belastet sie den Bus zusätzlich.

Es wird vereinbart, daß jedes Automatisierungssystem in bestimmten Zeitabständen (z.B. 60 s) mindestens ein Telegramm an die anderen Automatisierungssysteme sendet. Falls in diesem Zeitraum kein Sendeauftrag vorliegt, wird ein Sondertelegramm zum jeweiligen Automatisierungssystem abgeschickt. Falls der Empfänger ein benachbartes Automatisierungssystem ist, ist das Sender-Automatisierungssystem durch Auswertung der

internen Quittung des Empfänger-Automatisierungssystems in der Lage, jederzeit den aktuellen Verbindungszustand zu diesem Automatisierungssystem zu ermitteln. Bei Ausbleiben der Quittung wird registriert, daß dann im nächsten Zyklus kein Telegramm für das ausgefallene Automatisierungssystem mehr gebildet wird (Abbildung 7.27).

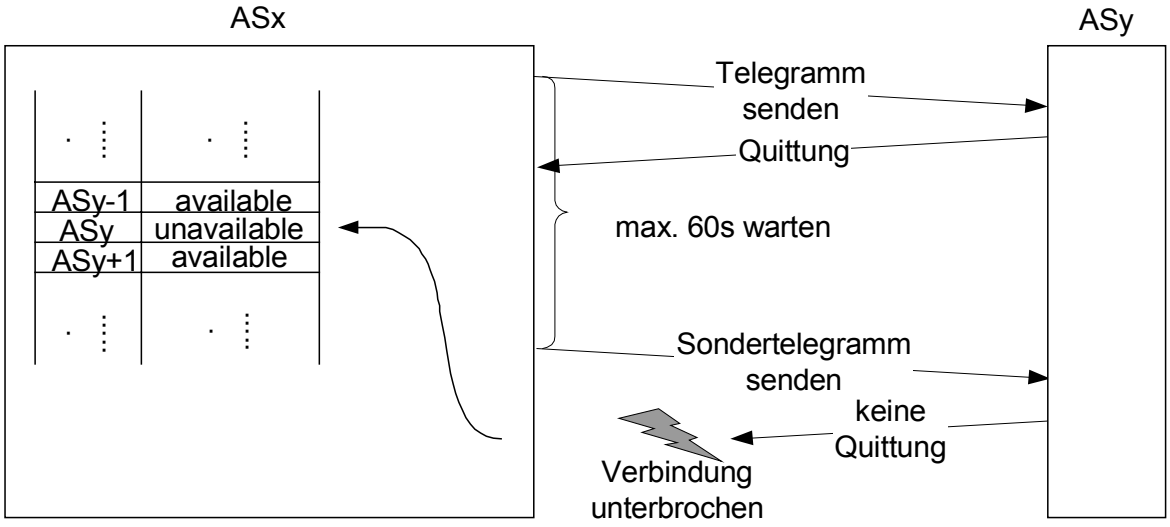


Abbildung 7.27: Verbindungsabbruch zwischen benachbarten Automatisierungssystemen

Falls zwischen Sender und Empfänger andere Automatisierungssysteme liegen, erfahren zuerst die dazwischen liegenden Automatisierungssysteme den Verbindungszustand zum Empfänger. Falls die Verbindung unterbrochen wird, schickt das erste Automatisierungssystem, das das Sondertelegramm vom Sender empfängt und den Verbindungszustand zum Empfänger weiß, eine negative Quittung zum Sender (Abbildung 7.28).

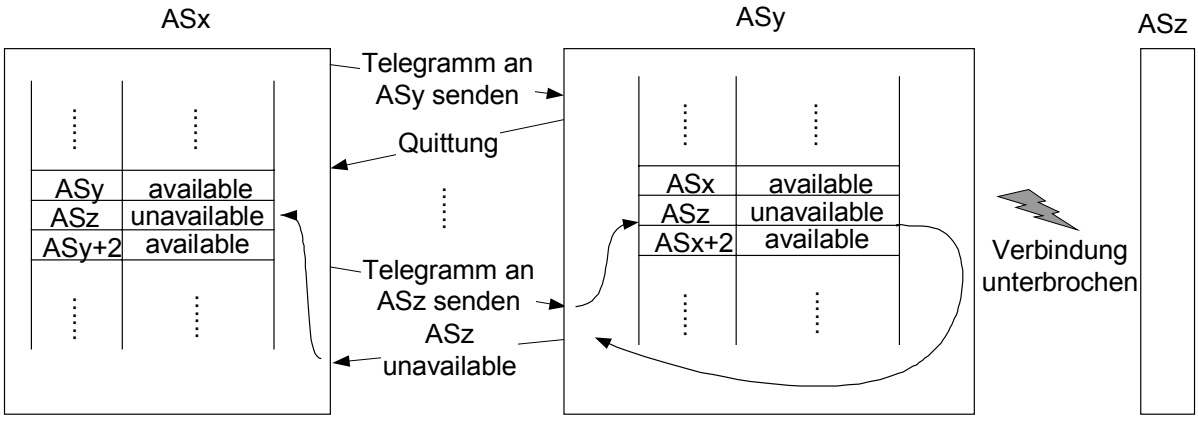


Abbildung 7.28: Verbindungsabbruch zwischen entfernten Automatisierungssystemen

Automatische Entdeckung einer neuen Verbindung

Es soll aber auch die Möglichkeit geben, den aktiven Teilnehmer wieder zu entdecken, falls ein ausgefallenes Automatisierungssystem wieder im Einsatz ist oder die Verbindung wieder eingerichtet wird.

Zu diesem Zweck bildet das im Betrieb befindende Automatisierungssystem ein Sondertelegramm, das keine Zieladresse beinhaltet. Das Telegramm wird in einem bestimmten Zeitabstand (z.B. 30 s) auf die Verbindung gelegt, wo ein anderer benachbarte Verbindungspartner vorher als "unerreichbar" registriert war. Falls das benachbarte Automatisierungssystem das Telegramm empfangen kann, wird eine Antwort dort gebildet und zum initiiertem Automatisierungssystem zurückgeschickt. Nachdem das initiierte Automatisierungssystem die Antwort erhalten hat, wird die Verbindung eingerichtet und als "erreichbar" registriert.

Der obengenannte Vorgang gilt nur für die Verbindungsüberwachung zwischen den Nachbarn. Um auch die entfernten Automatisierungssysteme miteinbeziehen zu können, wird eine Nachricht vom direkt betroffenen Nachbarn an alle Automatisierungssysteme geschickt, die diese Verbindung auch nutzen. In der Abbildung 7.29 wird ein Beispiel gegeben. Für die wiederhergestellte Verbindung zwischen AS 3 und AS 4 ist die Verbindung wieder einsatzbereit. In der ersten Phase wird die Nachricht zwischen den Nachbarn AS 3 und AS 4 ausgetauscht. In der zweiten Phase teilt AS 4 AS 18 mit, daß AS 3, AS 20, AS1 und AS 7 jetzt wieder erreichbar sind. Auf anderer Seite wissen AS 20, AS 1 und AS 7 wiederum von AS 3, daß AS 4 und AS 18 wieder aktiv sind.

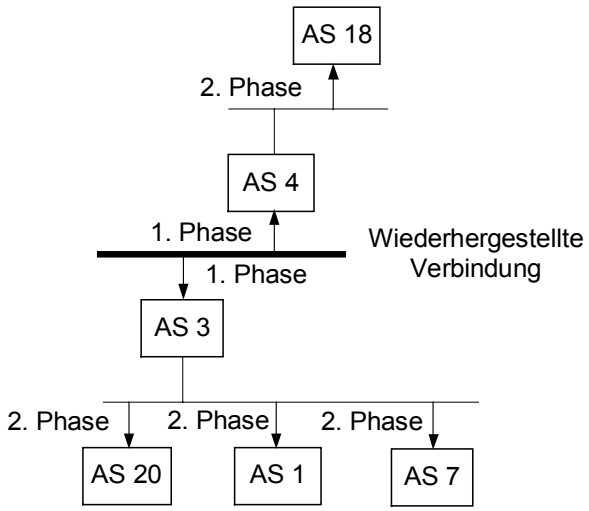


Abbildung 7.29: Automatische Entdeckung einer neuen Verbindung

7.7.3 Datensicherheit

Im Bereich Automatisierungstechnik wird eine hohe Datensicherheit angestrebt. Ein Kommunikationskonzept muß auch diesen Punkt berücksichtigen. BSEND/BRCV benutzt einen gesicherten, verbindungsorientierten Kommunikationsdienst. Die Datenübertragung

wird durch die Systemleistung intern überwacht. Das IK Modell bietet hier neue Möglichkeiten, die Datensicherheit bei der Übertragung zusätzlich zu erhöhen.

Einführung von Telegrammnummer

Ein Automatisierungssystem hat für jeden benachbarten Kommunikationspartner einen 4 Bit-Telegrammzähler erzeugt, der von 0 bis 15 zyklisch läuft. Der Zähler wird nach der Anlaufphase mit 0 initialisiert. Bei jeder Telegrammbildung wird der Zähler sich um 1 erhöhen und der Zählstand als Telegrammnummer im Telegrammkopf integriert. Nach der Ankunft des Telegramms wird durch die Telegrammnummer geprüft, ob das empfangene Telegramm der Nachfolger des vorhergehenden ist. Falls ja, wird das Telegramm angenommen. Im Fall einer Nichtübereinstimmung wird der Kommunikationspartner als "unerreichbar" registriert. Die Überprüfung der eingeführten Telegrammnummer bietet somit eine zusätzliche Methode, die Richtigkeit des Telegramms zu überwachen.

Extra Quittung auf der Anwendungsschicht

Die Einführung einer extra Quittung auf der Anwendungsschicht ist eine andere Möglichkeit zu prüfen, ob das Telegramm richtig angekommen ist. Jedes empfangene Telegramm wird nach der Auspackung extra durch ein Sondertelegramm quittiert. Das gleiche Telegramm wird wieder gesendet, falls die Quittung nach bestimmter Zeit nicht ankommt.

Nachsenden bei negativer Quittung vom Nachbarn

In den meisten Fällen wird der Empfangsbaustein in OB mit niedriger Priorität eingebaut. Falls das Automatisierungssystem wieder stark von anderen höherprioren Anwendungen belastet wird, kann es vorkommen, daß die für das Empfangen des Telegramms zuständige Routine relativ selten zum Einsatz kommt. Aber BSEND/BRCV sieht es vor, daß das Automatisierungssystem erst Daten empfangen darf, nachdem die vorherigen Daten abgeholt wurden. Es wird verweigert, die noch nicht gelesenen Daten zu überschreiben. Eine mögliche Folge ist dann die Zurückweisung von neu angekommenen Daten.

Um die dadurch verursachten Übertragungsfehler zu umgehen und die wichtigeren Prozeßdaten nicht zu verpassen, kann das Automatisierungssystem dasselbe Telegramm mehrfach senden, nachdem ein vorhergehendes Telegramm negativ quittiert wurde (Abbildung 7.30). Es ist nur von Interesse, wenn zwischen zwei unterschiedlichen Sendeaufträgen noch genügend Zeit bleibt.

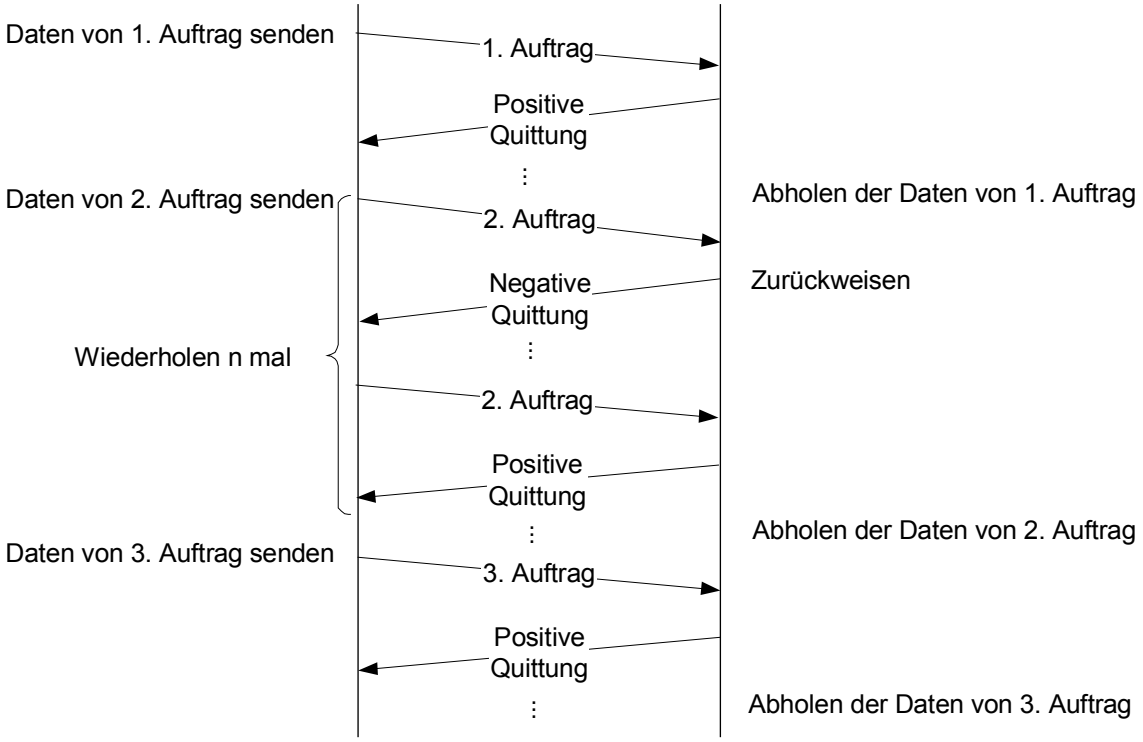


Abbildung 7.30: Nachsenden bei negativer Quittung

7.7.4 Busentlastung

Wie am Anfang dieser Kapitel erwähnt, werden die Daten bei den Konzepten in Kapitel 5 und 6 zyklisch übertragen, gleichgültig, ob eine Änderung vorliegt oder nicht. Falls der Kommunikationsbaustein in einem schnellen Zyklus aufgerufen wird und der zu übertragende Prozeßwert sich wiederum langsam ändert, wird das Telegramm mit gleichem Inhalt vielfach gesendet. Der Bus wird mit dieser wiederholten Übertragung des gleichen Inhalts unnötig belastet. Außerdem wird Rechenleistung für die Telegrammpackung und -entpackung verschwendet.

Beim IK Modell erfolgt die Übertragung ereignisgesteuert. Es wird nur dann ein Telegramm gebildet, falls eine Änderung des Prozeßwertes vorliegt. Die Überprüfung der Änderung nimmt zwar eine zusätzliche Rechenleistung in Anspruch, aber viel weniger als bei zyklischer Datenübertragung.

7.8 Meßergebnisse

Unter gleichen Bedingungen wie für die anderen Modelle wird auch das IK-Modell untersucht.

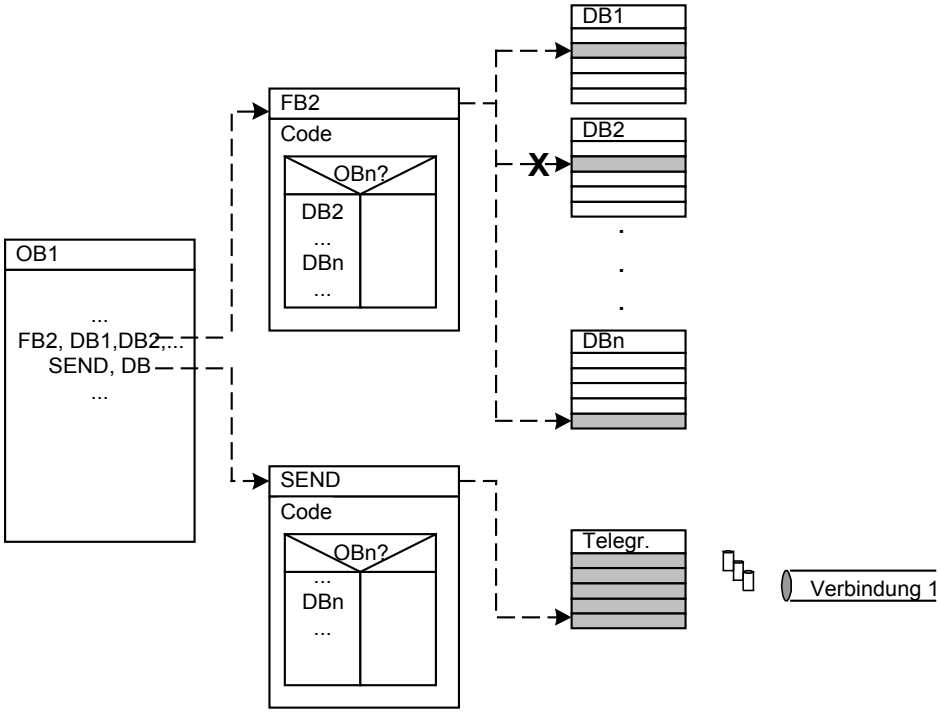


Abbildung 7.31: Bearbeitungsreihenfolge auf der Sendeseite (Implizite Methode)

Im Vergleich zu blockorientierter Übertragung ist hier ein zusätzlicher Rechenaufwand, der durch Fetchmechanismus und Telegrammbildung verursacht wird, auf dem Automatisierungssystem entstanden (Abbildung 7.31). Das drückt sich in einer längeren Rechenzeit aus, die die Kommunikationsaufträge benötigen. Die Performance wird somit negativ beeinflußt.

Wie bei blockorientierter Übertragung besteht hier die Rechenzeit aus zwei Teile. Ein Teil ist für einen Funktionsbaustein, der für die Organisation der Kommunikation zuständig ist. Und der andere Teil ist für den Sende- oder Empfangsbaustein. Die Rechenzeit auf der Sende- und Empfängerseite ist wie folgt,

$$t_{3S} = T_{3S} + T_{SEND}$$

$$t_{3R} = T_{3R} + T_{RCV}$$

Die tatsächliche Rechenzeit hängt hier von der Taktzeit der Signaländerung ab. T_{3S} und T_{3R} sind jeweils die Zeiten, die bei maximal erlaubter Signaländerung entstehen.

Im Gegensatz zur expliziten Methode hängt die Rechenzeit bei blockorientierter Übertragung und impliziter Methode nicht von der Anzahl der zu übertragenden Signale ab. Das heißt, ab bestimmte Signalanzahl bieten die beiden Methoden eine bessere Performance bezüglich der Rechenzeit (Abbildung 7.32).

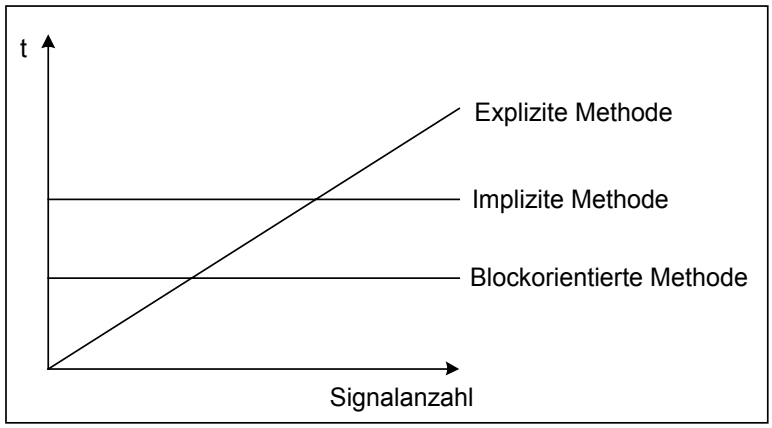


Abbildung 7.32: Bearbeitungszeit in Abhängigkeit von der Signalanzahl

Einfluß der Taktzeit

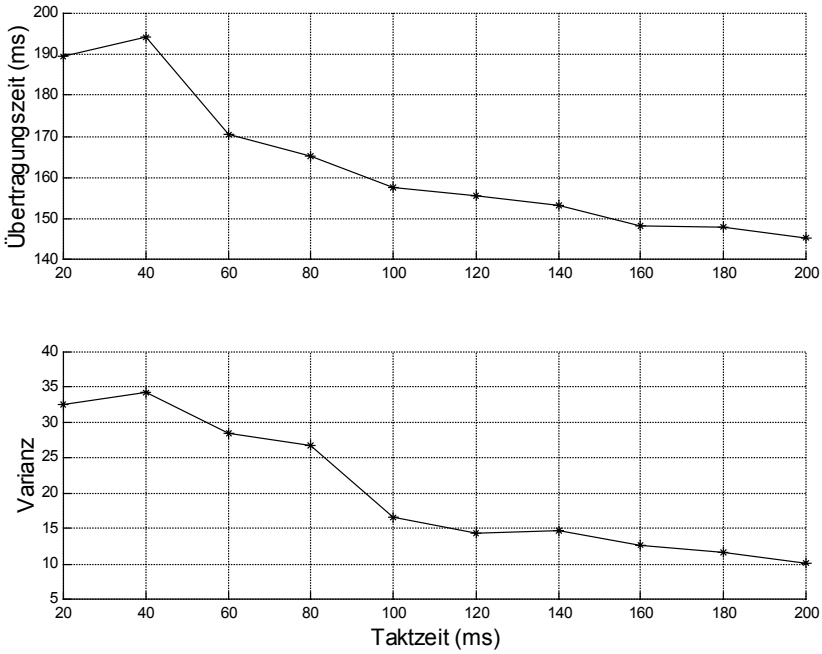


Abbildung 7.33: Übertragungszeit und Varianz in Abhängigkeit von der Taktzeit

Im Vergleich zu vorherigen Modellen ist die Übertragungszeit für das Einzelsignal beim IK-Modell stark angestiegen. Der Zeitzuwachs ist durch die längere Rechenzeit im OB1 begründet, die für den Mechanismus zur Telegrammbildung notwendig ist.

Bei einer kleineren Taktzeit ist die Änderung der zu übertragenden Werte und die damit verbundene Sendeanforderung häufiger. Da steigert sich auch die Rechenlast, die durch Fetchmechanismus und Telegrammbildung entstanden ist. Kommunikationsbausteine werden verspätet aufgerufen. Für Senden und Empfangen des Telegramms muß das Automatisierungssystem längere Zeit warten. Das bedeutet eine Verlängerung der

Reaktionszeit und verursacht auch eine große Spanne der Übertragungszeit, die eine größere Varianz bedeutet.

Als Beispiel wird in der Abbildung 7.34 eine Sendeanforderung mit einer Taktzeit von 100 ms und 200 ms gezeigt. Die Zeit, die OB1 für einen Kommunikationszyklus braucht, ist bei 200 ms kürzer als bei 100 ms.

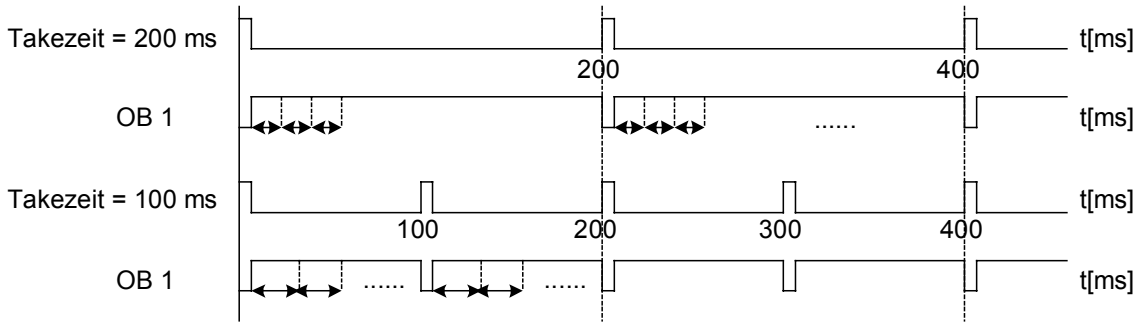


Abbildung 7.34: Zeitbedarf der Kommunikation in Abhängigkeit von der Taktzeit

Wie die Abbildung 7.35 zeigt, ist die Fehlerquote ist bei schnelleren Takten auch deutlich größer als bei dem anderen Modellen.

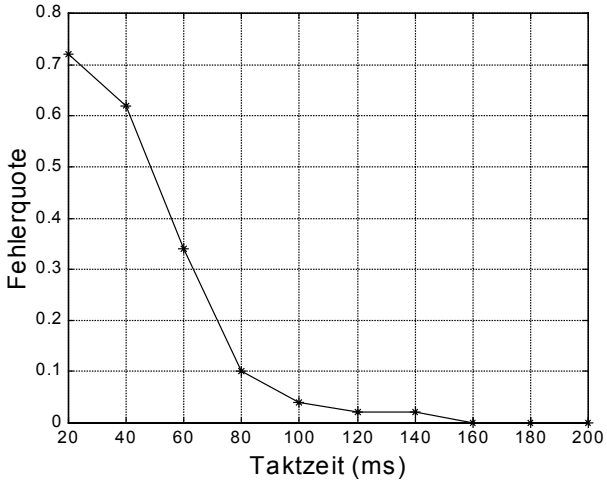


Abbildung 7.35: Fehlerquote in Abhängigkeit von der Taktzeit

Mit kürzerer Taktzeit ist die Fehlerrate steigend. Hier wird der Fehler auch teilweise verursacht durch die Zurückweisung des Datenpakets durch den Empfänger-AS, wie in Kapitel 5.5.1 schon beschrieben ist. Außerdem kommt hier noch eine Fehlerquelle hinzu, die aus anderer Sicht aber von Vorteil ist. Falls mehrere Signalquellen senden möchten, werden die Prozeßwerte zuerst zu einer Warteschlange geschickt. Je nach möglicher Telegrammgröße werden auf einmal mehrere Prozeßwerte in einem Telegramm zum Nachbar geschickt. Das reduziert die Busbelastung. Aber wenn das gleiche Signal sich so schnell ändert, daß der vorherigen Prozeßwert noch in der Warteschlange steht, wird der alte Wert verworfen. Er wird in der Messung als Fehler registriert.

Einfluß der Last

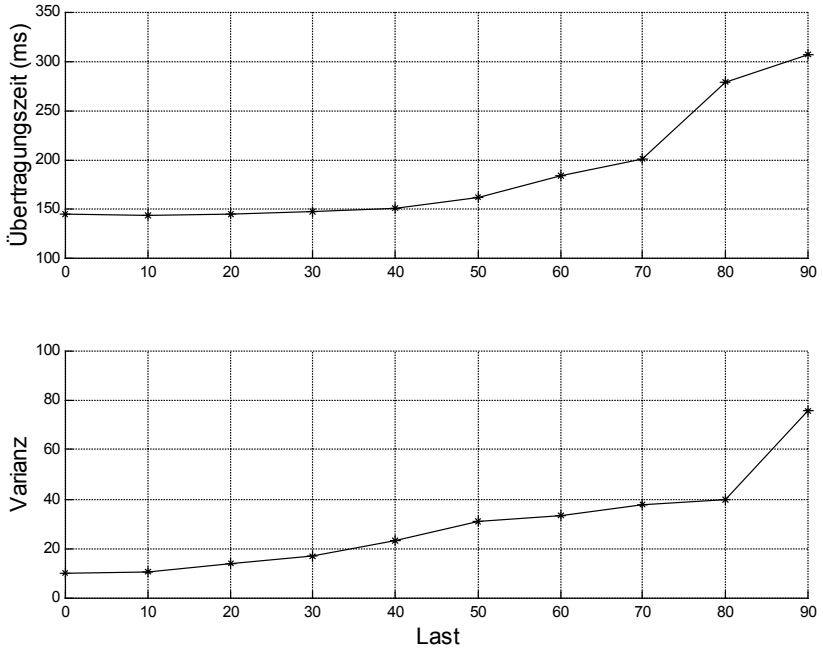


Abbildung 7.36: Übertragungszeit und Varianz in Abhängigkeit von der Last

Die Übertragungszeit und Varianz vergrößern sich auch hier mit steigender Last. Beim IK-Modell kommt der Übertragungsfehler ab Last = 60. Hinzu mit steigender Last steigt die Fehlerquote auch. Die Hauptursache ist, daß der Sendebaustein eine negative Quittung vom Empfänger-AS bekommt, weil der Kommunikationsbaustein nicht rechtzeitig aufgerufen wird. Diese Wahrscheinlichkeit wird sich erhöhen, wenn sich die Last erhöht.

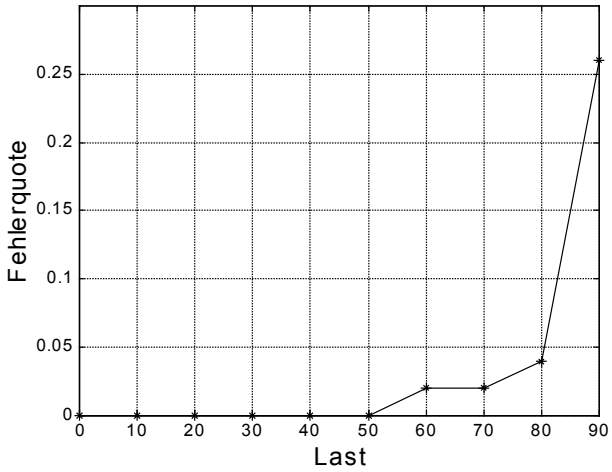


Abbildung 7.37: Fehlerquote in Abhängigkeit von der Last

Einfluß von der Anzahl der Blöcke

Beim IK-Modell kann mit Hilfe von "Blockung" ein großer Kommunikationsauftrag in mehrere kleinere Aufträge unterteilt werden. Diese Methode hat vor allem den Zweck, die Rechenleistung auch für andere Anwendungen im OB mit gleichen oder niedrigeren Prioritäten bereitzustellen. Sonst läuft man Gefahr, daß die Kommunikationsaufgabe die Ausführung anderer Anwendungen blockiert.

Die Abbildung 7.38 zeigt, wie eine Blockung funktioniert. Mit größer werdender Anzahl von Blöcken wird das Telegramm in mehrere kleinere Teiltelegramme zerlegt, die zu unterschiedlichen Zeitpunkten gebildet werden.

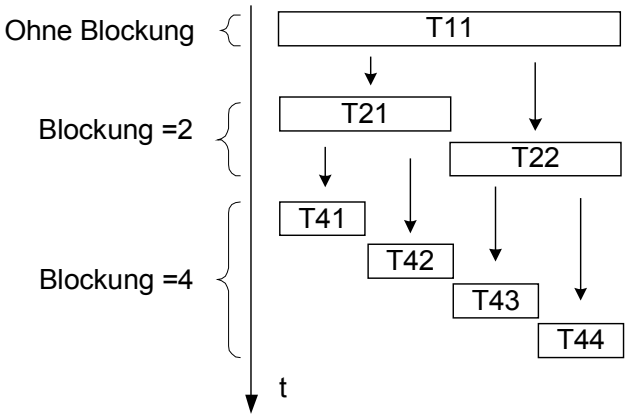


Abbildung 7.38: Blockung

In Test wird das gesamte Sendepaket jeweils mit Teilungsfaktor 1, 2, 3, 4 geteilt, wobei 1 ohne Teilung bedeutet. In der Abbildung 7.39 wird das Ergebnis in diskreter Form dargestellt.

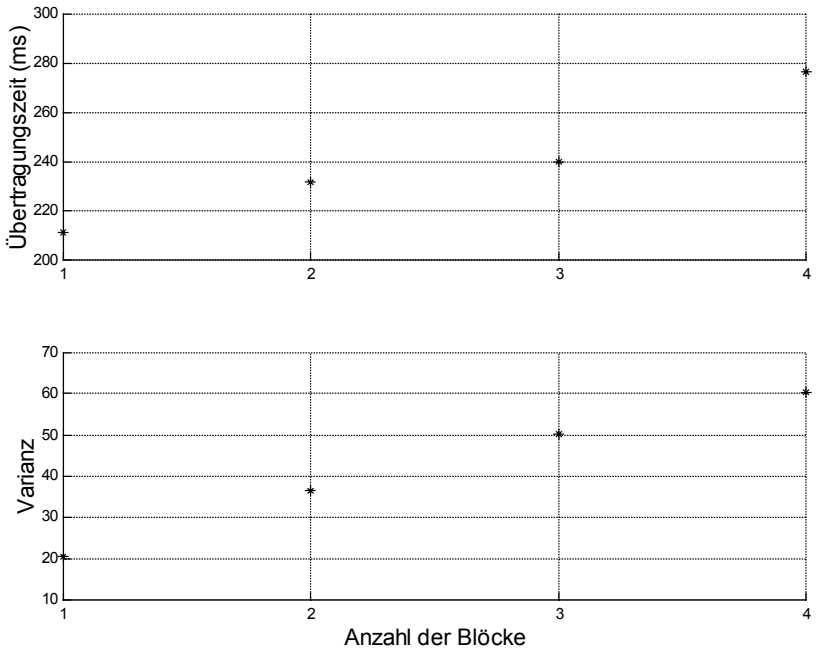


Abbildung 7.39: Übertragungszeit und Varianz in Abhängigkeit von der Anzahl der Blöcke

Mit obengenanntem Vorteil bringt die Blockung auch Performanceverlust mit sich. Mit kleiner werdender Anzahl der Blockung vergrößern sich Übertragungszeit und Varianz. Der Prozeßwert im Fetchstack wird nach der Reihenfolge in das Telegramm geschrieben. Ist die Blockgröße erreicht, müssen die Reste warten, bis der nächste Auftrag von der Telegrammbildung kommt. Das verlängert die Übertragungszeit und vergrößert die Varianz. Außerdem wird der Bus durch die Vielzahl von Telegrammen und das Automatisierungssystem durch mehrfach Telegrammbildung stärker belastet.

8 Vergleich der Kommunikationskonzepte

Die Kriterien, womit eine wirtschaftliche Nutzbarkeit eines Kommunikationskonzepts ausgewertet werden kann, wurden in der Kapitel 4 beschrieben. Sie können global aus unterschiedlichen Gesichtspunkten in drei Kategorien gegliedert werden:

- Performance,
- Kommunikationsmanagement,
- Benutzerfreundliche Projektierung (Usability).

8.1 Performance

Unter Performance wird in dieser Arbeit die Datenübertragungszeit, die Busbelastung sowie die Fehlerrate in Folge der Überlastung betrachtet.

Das Modell mit expliziter Verwendung spezifischer Kommunikationsbausteine realisiert eine zyklische Datenübertragung zwischen den Kommunikationsteilnehmern. Für ein Signal ist eine gute Performance zu erzielen, z.B. bezüglich der Übertragungszeit. Aber in der Praxis kommt es meistens vor, daß mehrere Signale zu übertragen sind. Mit immer größer werdender Anzahl der Signale verschlechtern sich die Performance signifikant, weil dafür immer mehr Kommunikationsbausteine aufgerufen werden müssen.

Das Modell mit blockorientierter Übertragung bietet auch eine zyklische Datenübertragung an. Es wird eine gute Performance gemessen. Die Anzahl der Signale hat hier wenig Einfluß auf die Performance, weil alle Prozeßwerte in einem Telegramm übertragen werden.

Eine ereignisgesteuerte Übertragung ist erstmals im IK-Modell durch die Verwendung des Fetchmechanismus möglich. Es werden nur Daten übertragen, wenn eine Änderung stattfindet. Die gleichgebliebenen Signale werden den Bus nicht mehr belasten. Somit ist eine Reduktion des Datenaufkommens erreicht. Es ist allerdings auf dem Automatisierungssystem mehr Grundlast entstanden. Die Konsequenz ist, daß im Automatisierungsgerät selbst für die Überwachung, Beurteilung und Vorverarbeitung der Signale ein großer Aufwand getrieben werden muß. Dieser geht dort mit einem deutlich vermehrten Speicherbedarf für Daten und Programmcode sowie zusätzlicher Laufzeit einher. Durch die Verzögerung der Ausführung der Kommunikationsaufgabe ist die Performance somit schlechter als beim Modell mit blockorientierter Übertragung. Es ist möglich, den Nachteil durch Einbindung der IK-Funktionalitäten in den unteren Kommunikationsschichten zu vermeiden. Aus gleichem Grund wie bei der blockorientierter Übertragung hat die Anzahl der Signale hier auch kaum Einfluß auf die Performance.

Aus dem Vergleich ist es ersichtlich, daß das Modell mit blockorientierter Übertragung und das IK-Modell jeweils eigene Vorteile haben. Die blockorientierte Übertragung weißt eine gute Performance auf, wobei das IK-Modell wiederum bei der Busentlastung große Vorzüge hat.

8.2 Kommunikationsmanagement

Im Vergleich zur expliziten Kommunikationsmethode bietet die blockorientierte Übertragung auch unterschiedliche Kommunikationsdienste bezüglich der Datensicherheit, z.B. gesicherter Dienst (BSEND/BRCV) und ungesicherter Dienst (USEND/URCV). Außerdem ist es möglich, die Kommunikationsaufgabe durch online einstellbare Takte durchführen zu lassen, z.B. schneller Dienst (100 ms) und langsamer Dienst (1 s).

Im Bezug auf Test und IBS-Phase wird bei der blockorientierten Übertragung eine Überwachungsfunktion eingeführt, mit der der Prozeß im laufenden Betrieb auf Fehler überwacht werden kann. Um einen sicheren Betrieb möglichst auch im Fehlerfall zu gewährleisten, wird hier eine "Force-Methode" mit Ersatz- und Defaultwert eingesetzt. Die Fehlerbehandlung ermöglicht einen Durchblick über Fehlerquelle, Fehlerart, usw., damit eine schnelle Reaktion durch den Anwender folgen kann.

Durch die Überwachung der anderen Netzelemente ist jedes Automatisierungssystem bei der Anwendung des IK-Modells in der Lage, ein neues Netzelement oder ein ausgefallenes Element zu erkennen. Danach wird die Kommunikationsverbindung eingerichtet oder abgebaut. Falls es Projektierungsinformationen gibt, die die ressourcenübergreifende Verbindung betreffen, erfolgt nach der Verbindungseinrichtung ein Informationsabgleich. Dabei muß es nicht zwingend sein, das der unveränderte Kommunikationspartner extra geladen werden muß. Die Mechanismen sollten auf Eigeninitiative online ihre Konfiguration nachbessern und auf den aktuellen Stand bringen.

Bisher müssen eine oder mehrere Verbindungen zwischen zwei Automatisierungssystemen für die Kommunikation reserviert werden. Einerseits werden die Verbindungen für den gegenseitigen Datenaustausch benutzt, wenn die Telegramme ausschließlich für diese beiden Automatisierungssysteme bestimmt sind. Andererseits werden die Verbindungen auch für die Kommunikation zwischen den Knoten reserviert, wobei die projektierten Automatisierungssysteme nur als "Übergänge" betrachtet werden. Ein Automatisierungssystem hat bisher nicht die Fähigkeit, die Daten anhand des Inhalts zu unterschiedlichen Nachbarn umzuleiten. Um den Projektierungs- und Ressourcenaufwand weitgehend zu reduzieren, spielt diese Fähigkeit eine immer wichtigere Rolle. Außerdem ist sie für eine Standardisierung der Kommunikation innerhalb der SPS-Welt auch von wesentlicher Bedeutung. Im IK-Modell wird dieses Problem gelöst dadurch, daß der Kommunikationsteilnehmer die Telegramme zum Ziel weiterleiten kann, ohne zusätzliche Kommunikationsressourcen zu beanspruchen. Die Folge ist ein reduzierter Aufwand der Verbindungsprojektierung und eine sparsame Verwendung der Kommunikationsressourcen.

Eine Integration der erwähnten Funktionalitäten bei der blockorientierten Übetragung ist im IK-Modell ebenfalls möglich. So ist ein kompakterer Funktionalitätsumfang beim IK-Modell zu erreichen.

8.3 Benutzerfreundliche Projektierung (Usability)

Ein immer wichtiger werdender Gesichtspunkt ist die effektivere Nutzung der Werkzeuge (Usability). Die Industrie fordert immer flexiblere und einfachere Projektierungsmethoden, um ständig sich ändernde und zunehmende Anforderungen erfüllen zu können. Das Ziel ist, die Automatisierungsaufgabe mit möglichst geringem Aufwand abzuwickeln.

Bei der expliziten Methode wird die Kommunikation durch Plazierung von externen Kommunikationsbausteinen eingerichtet. Die projektierten Parameter (ID, R-ID) in den Bausteinen bieten eine eindeutige Kommunikationsbeziehung zwischen den sendewilligen und empfangswilligen Bausteinen. Falls mehrere Signale gleichzeitig übertragen werden sollen, muß für jedes Signal oder eine bestimmte Anzahl von Signalen (bei Einführung von neuen Funktionen) ein BSEND/BRCV im Plan projektiert und parametriert werden. So ist der Zeitaufwand für die Kommunikationsprojektierung zwischen den Automatisierungssystemen erheblich. Bei der Änderungsprojektierung müssen alte Kommunikationsbausteine gelöscht, neue hinzugefügt und parametriert werden. Nach der Projektierung sind die Änderungen auf beide betroffene Automatisierungssysteme zu laden.

Die blockorientierte Übertragung bietet hier einen entscheidenden Fortschritt. Beim Erstellen der CFC / SFC - Pläne werden analog zu planlokalen Verschaltungen ressourcenübergreifende Verschaltungen projektiert. Ressourcenübergreifende Verschaltungen werden im Editor markiert, und ihre Eigenschaften können über Attribute projektiert werden (schnell, langsam, gesichert, ungesichert). Im Vergleich zur expliziten Methode erleichtert das Modell dem Anwender die Projektierung erheblich, weil die Verbindung anstatt durch externe Bausteine durch Verschaltung realisiert wird. Bei der Änderung ist dann nur eine Umschaltung vorzunehmen. Danach ist die Änderung auf beiden Seiten zu laden. Ein wichtiger Punkt ist, daß eine eindeutige Kommunikationsbeziehung hier durch die Position innerhalb des BK-Sende-DB und des BK-Empfangs-DB festgelegt ist. Um die Unabhängigkeit zwischen den Kommunikationsteilnehmern zu erreichen, ist eine invariante Pufferbelegung notwendig.

Beim IK Modell ist man auch in der Lage, Verbindung durch die Verschaltung zu projektieren. Der Projektierungszeitaufwand hat sich im Vergleich zur expliziten Methode um mindestens um Faktor 10 reduziert. Außerdem hat das IK Modell einen großen Vorteil bei der Änderungsprojektierung. Um der Leitstation die Änderungen rechtzeitig mitteilen und eine unnötige Rechen- und Kommunikationslast vermeiden zu können, wird jede Verbindung bzw. jeder Kommunikationsteilnehmer durch Sondertelegramme überwacht. Bei der Entdeckung wieder einsatzbereiter Kommunikationsteilnehmer werden alle Kommunikationsteilnehmer benachrichtigt, und es findet ein Informationsausgleich statt. Die Änderungsinformation, z.B. welche Verbindung hinzugekommen ist oder gelöscht wurde, wird dadurch automatisch aktualisiert. Für die Änderungsprojektierung braucht man deswegen nur einseitig zu laden, was eine erhebliche Erleichterung für weitere Planung der Projektierung bedeutet. Bei dem Ladevorgang ist völlig unproblematisch, in welchem Zustand (RUN, STOP, ...) die CPU sich gerade befindet. Im Bezug auf Usability bietet der IK-Modell somit den größten Komfort zu anderen Modellen.

8.4 Schlußfolgerung

Die explizite Methode bietet keinen Vorteil auf allen drei Gebieten. Sie läßt sich höchstens für kleine Projektpläne einsetzen. Dagegen kann man das Modell mit blockorientierter Übertragung und das IK-Modell auch für große Projektpläne benutzen. Das IK-Modell hat kompaktere Funktionalitäten auf dem Gebiet Kommunikationsmanagement und läßt sich leichter einsetzen im Bezug auf Usability.

Die blockorientierte Übertragung weist ein bessere Performance auf, wobei das IK-Modell wegen der höheren Grundlast etwas mehr Zeit braucht, wobei sich durch die Einbindung der Funktionalität in den unteren Kommunikationsschichten die Performance durchaus verbessern kann.

Insgesamt kann man feststellen, daß das neu entwickelte IK-Modell auf allen drei Gebieten ein zufriedenstellendes Ergebnis liefern kann. Deshalb soll das IK-Modell künftig für die Weiterentwicklung im Bezug auf die Kommunikation zwischen AS bevorzugt angewendet werden.

9 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurden Kommunikationskonzepte unter Berücksichtigung spezifischer Anforderungen und Merkmale in der Kommunikationswelt der Zellenebene hergeleitet, in Prototypen realisiert, getestet und bewertet.

Die explizite Methode der Kommunikation zwischen Automatisierungsgeräten realisiert die geräteübergreifende Verschaltung durch Einführung von externen Kommunikationsbausteinen. Der durch die Plazierung und Parametrierung entstandene Projektierungsaufwand ist hierbei relativ hoch. Deswegen ist die explizite Methode bei komplexeren Projekten oder mehrfachen Änderungen bezüglich der Kommunikation nicht zufriedenstellend.

Mit dem Model der blockorientierten Übertragung oder mit Hilfe des IK-Mechanismus ist die Möglichkeit gegeben, eine geräteübergreifende Verbindung durch einfache Verschaltung der technologischen Bausteine innerhalb eines Planes zu projektieren. Dafür bleibt dem Anwender die umständliche und fehleranfällige Verbindungsparametrierung des Kommunikationsbausteins erspart. Den Vorteil sieht man am deutlichsten bei der Änderungsprojektierung. Bei expliziter Verwendung spezifischer Kommunikationsbausteine müssen alte Kommunikationsbausteine gelöscht und die neuen hinzugefügt werden. Außerdem müssen die neuen Kommunikationsbausteine zusätzlich noch parametriert werden. Bei favorisiertem IK-Modell wird hingegen nur eine Umverschaltung der modifizierten Verbindung benötigt.

Beim Modell mit blockorientierter Übertragung wird die Möglichkeit gegeben, den Prozeß im laufenden Betrieb zu überwachen und die Ursache von Fehlern und Störungen im laufenden Betrieb zu erkennen. Mit der Ersatzwerte- oder einer Defaultwertstrategie wird ein relativ sicherer Betrieb gewährleistet, falls Probleme bei der Kommunikation auftauchen. Durch die Auswertung der Fehlerausgänge des Kommunikationsbausteins wird ein Kommunikationsfehler erkannt und kann vom Anwender behandelt werden. Durch Einführung von BK-FCs und BK-DBs ist ein Automatisierungssystem in der Lage, eine Fehlerhistorie aufzuzeichnen, damit eine nachträgliche Behandlung von zurückliegenden Fehlern möglich ist. Die hier vorgestellten Vorgehensweisen sind unabhängig vom jeweils benutzten Kommunikationsmodell und können in allen Konzepten eingesetzt werden.

Zur Verringerung der Busbelastung wird beim IK-Modell der Fetch-Mechanismus eingeführt, der eine ereignisgesteuerte Übertragung ermöglicht. Eine schnellere Versendung für die prozeßrelevanten Daten ist durch die Benutzung von unterschiedlich priorisierten FIFOs möglich. Die angestrebte Dezentralisierung der Information wird durch Einführung von Referenztabelle und Stacks realisiert. Dabei läuft nach dem Laden auf ein Automatisierungssystem ein Informationsabgleich per Sondertelegramm automatisch ab. Somit kann ein neuerliches Laden beim Kommunikationspartner vermieden werden. Das bringt eine große Erleichterung bei der Änderungsprojektierung (z.B. bei einem sternförmigen Netzwerk).

Beim Modell der Impliziten Kommunikation werden Telegramme in der Anwendungsschicht gebildet. Es bietet mehr Möglichkeiten, den Datenaustausch zu beobachten oder zu kontrollieren. Durch Kontrolle der Adreßinformation in den Telegrammen ist jeder

Kommunikationsteilnehmer in der Lage, den Datenfluß zum richtigen Nachbar weiterzuleiten, bis die Daten beim vorgesehenen Ziel ankommen. Das "routingfähige" Automatisierungssystem erleichtert die Verbindungsprojektierung, weil nur die direkt zum Nachbarn führenden Verbindungen projektiert werden müssen. Hierdurch werden Kommunikationsressourcen auch weniger belastet.

Das IK-Modell überwacht den Zustand der Verbindung ständig. Neu hinzugefügte Kommunikationsteilnehmer werden vom Nachbarn automatisch erkannt. Bricht eine Verbindung zusammen, wird die Versendung von Daten ausgesetzt und kann bei geeigneten Sendepuffern ohne Datenverlust und Irritation der Anwendungsebene nachgeholt werden, wenn die Verbindung wieder hergestellt werden konnte.

Eine gute Performance kann bei blockorientierter Übertragung erzielt werden, wobei das IK-Modell hier zunächst geringe Nachteile aufweist. Die systemnahe Implementierung sollte diesen Nachteil überkompensieren können.

Nach der Auswertung aus verschiedenen Gesichtspunkten liefert das IK-Modell insgesamt ein besseres Resultat. Es reduziert den Aufwand sowohl bei der Projektierung als auch bei der Änderungsprojektierung. Eine Minderung der Busbelastung und eine Überwachung der Kommunikationsteilnehmer sind ebenfalls erreicht worden.

Ausblick

Bisher wurden ausschließlich homogene S7-Netze betrachtet. In der Weiterentwicklung der dargestellten Konzepte zur Kommunikation in der Zellenebene kann man sich vorstellen, eine Erweiterung der impliziten Kommunikation auf eine heterogene SIMATIC-Umgebung vorzunehmen. Dies beinhaltet hauptsächlich folgende Bereiche:

- Heterogene S7-S5-Netze,
- weitere Netzprotokolle wie

  FMS OBK (offene Bausteinkommunikation),

  FDL Hantierungsbaustein S5-Profibus,

  ISO Hantierungsbaustein S5-Industrial Ethernet.

Ein möglicher Lösungsansatz ist eine Erweiterung des nutzbaren Protokolls von dem Kommunikationsbaustein. So kann zwischen unterschiedlichen Kommunikationsbausteinen innerhalb des heterogenen Netzes eine beliebige Kommunikationsverbindung eingerichtet werden. Hier behandelte Kommunikationskonzepte basieren auf der oberen Kommunikationsschicht. Es gibt somit kein Hindernis, die Konzepte dort anzuwenden.

Für den Datenaustausch zwischen Automatisierungssystem und Operationssystem, wobei z.B. die Prozeßwerte beobachtet oder gesteuert werden sollen, kann das Modell der impliziten Kommunikation ebenfalls eingesetzt werden.

Literaturverzeichnis

/Abe90/ Abeln, O.
Die CA...-Techniken in der industriellen Praxis
Hanser Verlag (1990)

/Ahr87/ Ahrens, W.
Einsatz von Expertensystemen in der Prozeßleittechnik
Automatisierungstechnische Praxis 29 (1987)

/Ahr89/ Ahrens, W.; Polke, M.
Netzmodelle als systemtecchnische Informationsbasis für die Prozeßleittechnik: Der zukünftige computerunterstützte Ingenieurarbeitsplatz
Automatisierungstechnik 37 (1989)

/Ahr95/ Ahrens, W.; Spohr, G. U.
Auf dem Weg zu einer informationsorientierten Prozeßleittechnik
Automatisierungstechnische Praxis 37 (1995)

/Ahr97/ Ahrens, W.; Scheurlen, H.J.; Spohr, G.U.
Informationsorientierte Leittechnik
R. Oldenbourg Verlag, München, Wien (1997)

/Ald97/ Alder, J.
Technische Prozeßanalyse, Projektierung und Programmierung von Steuerungen
GMA-Fachtagung, VDI Verlag, Düsseldorf (1997)

/Anw98/ Anweisungsliste für S7-300/400, Referenzhandbuch
Siemens (1998)

/Arn99/ Arnold, M.
Kommunikationskonzept für die Prozeßleittechnik
Verlag Mainz, Wissenschaftsverlag, Aachen (1999)

/Aue92/ Auer, K.
PLS-spezifische CAE-Werkzeuge zur Planung der Prozeßautomatisierung
CAE-Werkzeuge zur Planung der Prozeßautomatisierung, VDE-Verlag (1992)

/Aue94/ Auer, A.
Steuerungstechnik und Synthese von SPS-Programmen
Hütig Verlag (1994)

/Aut99/ Automatisierungssysteme
S7-400, M7-400 Aufbauen, Installationshandbuch
Siemens (1999)

/Bal90/ Balzert, H.
Die Entwicklung von Software-Systemen. Prinzipien, Methoden, Sprachen, Werkzeuge
Wissenschaftsverlag (1990)

/Ben90/ Bender, K.
PROFIBUS
Carl Hanser Verlag, München, Wien (1990)

/BerJ99/ Bergmann, J.
Automatisierungs- und Prozeßleittechnik
Carl Hanser Verlag, München, Wien (1999)

/BerH99/ Berger, H.
Automatisieren mit SIMATIC
Publicis MCD Verlag, München (1999)

/Beu94/ Beuschel, J.
Prozeßsteuerungssysteme
R. Oldenbourg Verlag, München, Wien (1994)

/Buc86/ Buchner, K. H.
Technische Kommunikation in der Automatisierungstechnik
Automatisierungstechnische Praxis 28 (1986)

/CFC98/ CFC für S7
Continuous Function Chart, Handbuch
Siemens (1997)

/Cha00/ Chan, W. C.
PERFORMANCE ANALYSIS OF TELECOMMUNICATIONS AND LOCAL AREA NETWORKS
Kluwer Academic Publishers, Boston/Dordrecht/London (2000)

/Dan95/ Dannapel, B.; Hensel, H.; Möckel, B.; Mühlenkamp, J.; Müller-Heinzerling, Th.; Otto, A.; Teuchert, V.
Qualifizierung von Leitsystemen: Ein Gemeinschaftprojekt von GMA und NAMUR zur Validierung
Automatisierungstechnische Praxis 37 (1995)

/DIN 19226/ Regelungstechnik und Steuerungstechnik
Beuth Verlag (1994)

/Com99/ PCS 7 Communication Blocks, Handbuch
Siemens (1999)

/Die94/ Dietrich, H.; Hayka, H.; Jansen, H.; Kehrer, B.
Systemarchitektur des CAD-Referenzmodells unter den Aspekten Kommunikation, Produktdatenmanagement und Integration
GI-Fachtagung, Hanser Verlag (1994)

/Die96/ Dieterle, W.
Effiziente Kommunikations-Architekturen zum Aufbau verteilter Echtzeit-Datenbanken in industriellen Leitsystemen
VDI Verlag, Düsseldorf (1996)

/EN 61131-1/ DIN EN 61131-1
Speicherprogrammierbare Steuerungen, Teil 1
Beuth Verlag (1994)

/EN 61131-3/ DIN EN 61131-3
Speicherprogrammierbare Steuerungen, Teil 3
Beuth Verlag (1994)

/Fas93/ Fasbender, A.
Leistungsbewertung eines prioritätengesteuerten Realzeit-Kommunikationssystems
Kommunikation in verteilten Systemen, Springer Verlag, München (1993)

/Fär87/ Färber, G.
Bussysteme
R. Oldenbourg Verlag, München, Wien (1987)

/Fär94/ Färber, G.
Prozeßrechentechnik
Springer Verlag Berlin Heidelberg New York (1994)

/Fei97/ Feindt, E. G.
Entwurf und Simulation industrieller Steuerungen für den PC und die SPS
R. Oldenbourg Verlag, München, Wien (1997)

/Fil94/ Filbert, D.
Messen Elektrischer Größen
Vorlesungsunterlage, TU Berlin (1994)

/Fri85/ Friedl, A.; Kreppel, H.
Kommunikationsanforderungen in der Produktionsautomatisierung
Automatisierungstechnische Praxis 27 (1985)

/Frü97/ Früh, K. F.
Handbuch der Prozeßautomatisierung
R. Oldenbourg Verlag, München, Wien (1997)

/Gro94/ Gronau, N.
Computer Aided Manufacturing
CIM-Management, 1/94, R. Oldenbourg Verlag, München, Wien (1994)

/Gro94/ Gronau, N.
Steuerungsstrukturen in der Fertigung
CIM-Management, 4/94, R. Oldenbourg Verlag, München, Wien (1994)

/Gro94/ Gronau, N.
Kommunikation in der computerunterstützten Fertigung
CIM-Management, 5/94, R. Oldenbourg Verlag, München, Wien (1994)

/Grö96/ Grötsch, E.
SPS 1 – Speicherprogrammierbare Steuerungen, Einführung und Übersicht
R. Oldenbourg Verlag, München, Wien (1996)

/Grö97/ Grötsch, E.
SPS 2 – Speicherprogrammierbare Steuerungen, Programmbeispiele und Produkte
R. Oldenbourg Verlag, München, Wien (1997)

/Hat87/ Hatley, D. J.; Pirbhai, I. A.
Strategies for Real-Time System Specification
Dorset House PUblishing (1987)

/Her86/ Herrtwich, R. G.
Koordination in lokalen Netzen

Automatisierungstechnische Praxis 28 (1986)

/Her92/ Hermanns, O.
Ein offenes Kommunikationskonzept zur Vernetzung von CAD-Systemen
VDI- Fachberichte 993.3, VDI Verlag, Aachen (1992)

/Her95/ Hermanns, O.
Erfahrungen mit einer OSI-basierten Infrastruktur für heterogene verteilte Systeme
Proceedings zur KiVS '95, Springer Verlag, Chemnitz (1995)

/Hun96/ Hunt, C.
TCP/IP
O'Reilly/International Thomson Verlag, Bonn (1996)

/Jan85/ Janetzky, D.
Serielle Bussysteme in der Prozeßrechner- und Automatisierungstechnik
VDI Verlag (1985)

/Jäg91/ Jäger, K. W.
Schnittstellen bei CAD/CAE-Systemen
VDI Verlag (1991)

/Kaf98/ Kaftan, J.
SPS-Grundkurs mit SIMATIC S7
Vogel Buchverlag, Würzburg (1998)

/Kat89/ Katz, M.; Biwer, G.; Bender, K.
Die PROFIBUS-Anwendungsschicht
Automatisierungstechnische Praxis 31 (1989)

/Ker93/ Kerner, H.
Rechnernetze nach OSI
Addison-Wesley Publishing Company, Bonn (1993)

/Kir98/ Kirstädter, A.
Verteilte Koordinierungsverfahren für ATM-Vermittlungen mit Eingangspuffern
Herbert Utz Verlag Wissenschaft, München (1998)

/Kli93/ Klinker, W.
atp-Marktanalyse Speicherprogrammierbare Steuerungen
Automatisierungstechnische Praxis 35 (1993)

/Kli95/ Klinker, W.
atp-Marktanalyse "SPS-basierte Leitsysteme"
Teil 2: Tabellarische Übersicht
Automatisierungstechnische Praxis 37 (1995)

/Kom99/ Kommunikation mit SIMATIC, Handbuch
Siemens (1999)

/Kon98/ Konhäuser, W.
Industrielle Steuerungstechnik
Carl Hanser Verlag, München, Wien (1998)

/Kör87/ Körner, W.; Polke, M., Moll, P.

Zur funktionalen Gliederung von Leitsystemen
Automatisierungstechnische Praxis 29 (1987)

/Kru90/ Krückeberg, F.; Spaniol, O.
Lexikon Informatik und Kommunikationstechnik
VDI Verlag, Düsseldorf (1990)

/Her95/ Hermanns, O.
Erfahrungen mit einer OSI-basierten Infrastruktur für heterogene verteilte Systeme
Proc. of KiVS '95, Chemnitz (1995)

/Hod98/ Hodgkinson, T.
Internet and asynchronous transfer mode networking
The Institution of Electrical Engineers (1998)

/Hom86/ Hommel, G.; Schindler, S.
Informatik-Anwendungen – Trends und Perspektiven
GI Jahrestagung Bd. 16, Springer Verlag, Berlin (1986)

/Hom91/ Hommel, G.
Automatisierungs- und Leitsysteme in den neunziger Jahren
Prozessrechensysteme '91, Springer Verlag, Berlin (1991)

/ISO/ International Organisation for Standards
"Open Systems Interconnection Management Framework"
ISO-7498-4 (International Standard)

/Joh95/ John, K. H.; Tiegelkamp, M.
SPS-Programmierung mit IEC1131-3
Springer Verlag Berlin Heidelberg New York (1995)

/Lau99/ Lauber, R.; Göhner, P.
Prozessautomatisierung
Springer Verlag Berlin Heidelberg New York (1999)

/Leh91/ Lehmann, A.; Lehmann, F.
Messung, Modellierung und Bewertung von Rechnersystemen
6. GI/ITG-Fachtagung, Springer Verlag, Neubiberg (1991)

/Leh95/ Lehmann, S. K.
Ein objekt-orientiertes CAE-System
Automatisierungstechnische Praxis 37 (1995)

/Lew95/ Lewis, R. W.
Programming industrial control systems using IEC1131-3
IEE Control Engineering Series 50
The Institution of Electrical Engineers, London (1995)

/Lit99/ Litz, L.; Frey, G.
Methoden und Werkzeuge zum industriellen Steuerungsentwurf
Automatisierungstechnik, 4/99, R. Oldenbourg Verlag, München, Wien (1999)

/Mai95/ Maier, U.; Tauchnitz, T.
atp-Marktanalyse "SPS-basierte Leitsysteme"
Teil 1: Einordnung und Systemeigenschaften

Automatisierungstechnische Praxis 37 (1995)

/Mar00/ Martin, M.
Beitrag zur Generierung modularer, bausteingestützter Software für die Steuerung von Maschinen und Anlagen
VDI Verlag, Düsseldorf (2000)

/Mey94/ Meyer,B.; Popien, C.
Defining Policies for Performance Management in Open Distributed Systems
Proceedings of 5th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM'94), Toulouse (1994)

/Nau95/ Naunin, D.
Industrielle Steuerungstechnik
Vorlesung, Elektrotechnik, TU Berlin

/Neu95/ Neumann, P.; Grötsch, E.; Lubkoll, C.; Simon, R.
SPS-Standard: IEC1131
R. Oldenbourg Verlag, München, Wien (1995)

/Nol94/ Noll, P.
Nachrichten Übertragung I
Vorlesungsunterlage, TU Berlin (1994)

/Oht98/ Ohta, M.; Crowcroft, J.
Static Internet Multicast
The Institution of Electrical Engineers (1998)

/PCS97/ PCS7 Engineering System, Handbuch
Siemens (1997)

/Pim90/ Pimentel, J.
Communication Networks for Manufacturing
Prentice Hall, Englewood Cliffs (1990)

/Pol94/ Polke, M.
Prozeßleittechnik
R. Oldenbourg Verlag, München, Wien (1994)

/Pro89/ PROFIBUS-Norm DIN 19245 Teil 1
Beuth-Verlag, Berlin (1989)

/Pro90/ PROFIBUS-Norm DIN 19245 Teil 2
Beuth-Verlag, Berlin (1990)

/Pro96/ Process Control System PCS7
Systembeschreibung, Handbuch
Siemens (1996)

/Pro99/ PROFIBUS-Netze, Handbuch
Siemens (1999)

/Rau97/ Rausch, M.
Modulare Modellbildung, Synthese und Codegenerierung ereignisdiskreter Steuerungssysteme
VDI Verlag, Düsseldorf (1997)

/Rem79/ Rembold, U.
Prozeß- und Mikrorechnersysteme
R. Oldenbourg Verlag, München, Wien (1979)

/Rem94/ Rembold, U.; Levi, P.
Realzeitsysteme zur Prozeßautomatisierung
Carl Hanser Verlag, München, Wien (1994)

/SchE93/ Schnieder, E.
Prozeßinformatik
Vieweg und Sohn (1993)

/SchE94/ Schnieder, E.
Modellbildung im Übergang zur informationsorientierten Automatisierungstechnik
28. Regelungstechnisches Kolloquium, Boppard (1994)

/SchH95/ Schuler, H.
"Prozeßführung" – "Prozeßleittechnik" – "Prozeßautomatisierung"
Automatisierungstechnische Praxis 37 (1995)

/SchM94/ Schuba, M.; Hermanns, O.
Modellierung von Multicast
Aachener Beiträge zur Informatik, Bd. 7, (1994)

/SchG99/ Schnell, G.
Bussysteme in der Automatisierungstechnik
Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig (1999)

/SchR95/ Scheuring, R.
Modellierung, Beobachtung und Steuerung ereignisorientierter verfahrenstechnischer Systeme
VDI Verlag, Düsseldorf (1995)

/SchR99/ Schraft, R. D.; Kaun, R.
Automatisierung. Stand der Technik. Defizite und Trends in der Automatisierungstechnik
Verlagsgruppe Handelsblatt GmbH, WirtschaftsWoche, Düsseldorf (1999)

/SFC97/ SFC Sequential Function Chart, Handbuch
Siemens (1997)

/Spa91/ Spaniol, O.; Schümmer, M.
Netze in der Produktion
CIM-Management, 1/91, R. Oldenbourg Verlag, München, Wien (1991)

/Spo90/ Spohr, G. U.
CAE in der Prozeßleittechnik
INTERKAMA 89, Automatisierungstechnische Praxis 32 (1990)

/Spo92/ Spohr, G. U.
Anforderungsprofil und Möglichkeiten der CAE-Werkzeuge
CAE-Werkzeuge zur Planung der Prozeßautomatisierung
VDE Verlag (1992)

/Str88/ Strickling, A.
CAE in der Prozeßleittechnik, Einführung und Übersicht
Automatisierungstechnische Praxis 30 (1988)

/Str92/ Strohrmann, G.
Automatisierungstechnik
R. Oldenbourg Verlag, München, Wien (1992)

/Sys199/ Systemsoftware für S7-300/400
System- und Standardfunktionen, Referenzhandbuch
Siemens (1999)

/Sys299/ Systemsoftware für S7-300/400
Standardfunktionen Teil 2, Referenzhandbuch
Siemens (1999)

/Tan97/ Tanenbaum Andrew, S.
Computernetzwerke
Prentice Hall, München (1997)

/Tyc99/ Tyczynski, T.
SPS – Einsatz in der Gebäudetechnik
Verlag Technik, Berlin (1999)

/VDE86/ Stand und Entwicklung der Prozeßleittechnik
VDE Verlag, Berlin (1986)

/VDE90/ Speicherprogrammierbare Steuerungen in der Prozeßleittechnik
VDE Verlag, Berlin (1990)

/VDE91/ Künftige Anforderungen an die Prozeßleittechnik
VDE Verlag, Berlin (1991)

/VDE91/ SPS-gestützte Leitsysteme
VDE Verlag, Berlin (1997)

/VDI93/ Automatisierungstechnik, Sensoren/Aktoren, Leittechnik, Engineering,
Ganzheitliche Strategien
VDI Berichte 1067, GMA-Fachtagung, VDI Verlag, Düsseldorf (1993)

/VDI94/ Vernetzung durch industrielle Kommunikation
VDI Berichte 1123, GMA-Fachtagung, VDI Verlag, Düsseldorf (1994)

/Wal98/ Walker, P.J.; Harris, M.
A regulatory view of acces control
The Institution of Electrical Engineers (1998)

/Weh96/ Wehler, A.
Verteilte Automationssysteme mit Integration von Insellösungen
VDI Verlag, Düsseldorf (1996)

/Wei00/ Weigmann, J.; Kilian, G.
Dezentralisieren mit PROFIBUS-DP
Publicis MCD Verlag, München (2000)

/Zin93/ Zinser, K.

Neue Formen und Medien der Prozeßvisualisierung
Automatisierungstechnische Praxis 35 (1993)

/Zit97/ Zitterbart, M.
Kommunikation in verteilten Systemen
GI/ITG-Fachtagung, Springer Verlag, Braunschweig (1997)

Lebenslauf

Name:	Qimin Zhang
Geburtsdatum:	5. September 1971
Geburtsort:	Shanghai/VR China
Eltern:	Vater, Shouyi Zhang, Prof. Dr.-Ing. Mutter, Youchen Lin, Dipl.-Ing.
Familienstand:	verheiratet
Schulbildung:	1977 - 1989 Abitur in Hangzhou/China
Studium:	1992 – 1998 Studium der Elektrotechnik an der TU-Berlin Vertiefungsfach: Meß- und Automatisierungstechnik
Berufstätigkeit:	1990 – 1992 Praktikum in der Elektromaschinenbaufabrik der Zhejiang Universität und ABB Bergmann-Borsig GmbH 1996 – 1997 Werkstudent im Grundlagenforschungsprojekt „Nonlinear Adaptive Control Technology“ bei Daimler Benz AG in Berlin 1998 - 2000 Siemens AG, Karlsruhe Seit 2000: Lucent Technologies, Nürnberg