SIEMENS

SIMATIC S5

CPU 928B

Programmieranleitung

Bestell-Nr.: 6ES5 998-2PR11
Ausgabe 01

Tabellenheft CPU 922/CPU 928/CPU 928B/CPU 948

Bestell-Nr.: 6ES5 997-3UA12

ist dem Handbuch beigefügt

C79000-H8500-C898-01

Copyright

Haftungausschluß

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

Technische Änderungen bleiben vorbehalten.

Sicherheitstechnische Hinweise

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise sind durch ein Warndreieck hervorgehoben und je nach Gefährdungsgrad folgendermaßen dargestellt:

Vorsicht
Bedeutet, daß eine leichte Körperverletzung oder ein Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Inbetriebsetzung und Betrieb des Gerätes darf nur von **qualifiziertem Personal** vorgenommen werden. Qualifiziertes Personal im Sinne der sicherheitstechnischen Hinweise dieses Handbuchs sind Personen, die die Berechtigung haben, Geräte, Systeme und Stromkreise gemäß den Standards der Sicherheitstechnik in Betrieb zu nehmen, zu erden und zu kennzeichnen.

Siemens Aktiengesellschaft

6ES5 998-2 PR11
EWK Elektronikwerk Karlsruhe

Printed in the Federal Republic of Germany

Hinweise zur Benutzung des Handbuches

Geltungsbereich

Diese Programmieranleitung beschreibt die Funktionen der CPU 928B-3UB11 und CPU 928B-3UB12 sowie die zugehörige Systemsoftware.

Die zusätzlichen Funktionen der CPU 928B-3UB12 sind im Handbuch gekennzeichnet. Sie können zum Teil bei der CPU 928B-3UB11 nachgerüstet werden (lesen Sie hierzu bitte den Abschnitt 1.8).

Hinweise zu den Kapitel-Inhalten

Kapitel 1

informiert Sie über den Anwendungsbereich des Automatisierungsgerätes S5-135U mit der CPU 928B.
Es erklärt die typische die Arbeitsweise einer CPU und schildert, wie ein CPU-Programm aufgebaut ist.
Sie erhalten ferner einige Vorschläge, wie Sie beim Programmieren vorgehen können und erfahren, welche für die Programmierung wichtigen Kenndaten die CPU 928B hat.
Wenn Sie bereits mit der CP 928B-3UB11 gearbeitet haben und wissen möchten, welche Unterschiede die CPU 928B-3UB12 dazu hat, können Sie dies dort nachlesen.

Kapitel 2

schildert Ihnen, aus welchen Komponenten sich ein STEP-5-Anwenderprogramm zusammensetzt und wie es strukturiert werden kann.

Kapitel 3

wendet sich an Leser, die in der Anwendung der Programmiersprache STEP 5 noch keine große Erfahrung haben. Es führt daher in die Gundlagen der STEP-5-Programmierung ein und erläutert im weiteren ausführlich (mit Beispielen) die STEP-5-Operationen.

Erfahrenen Lesern, denen die Information zu einer speziellen STEP-5-Operation im Tabellenheft nicht ausreicht, kann der Abschnitt 3.5 als Nachschlageteil dienen.

Kapitel 4

gibt Ihnen eine Übersicht über die Betriebszustände und die Programmbearbeitungsebenen der CPU 928B. Es informiert Sie ausführlich über verschiedene Anlaufarten und damit verbundene Organisationsbausteine, in denen Sie Ihr spezifisches Programm für verschiedene Anlauffälle programmieren können.

Sie erfahren ferner, wodurch sich die Programmbearbeitungsebenen "Zyklische Bearbeitung", "Zeitgesteuerte Bearbeitung" und "Alarmgesteuerte Bearbeitung" auszeichnen und welche Bausteine für Ihr Anwenderprogramm dabei zur Verfügung stehen.

Kapitel 5

informiert Sie, wie Sie Fehler beim Planen und Programmieren Ihrer STEP-5-Programme vermeiden können.
Sie erfahren, welche Hilfen Ihnen das Systemprogramm zur Fehlerdiagnose und evtl. Fehlerreaktion zur Verfügung stellt und in welchen Bausteinen Sie Reaktionen auf bestimmte Fehler progammieren können.

Kapitel 6 führt die integrierten Sonderfunktionen das Systemprogramm auf. Es informiert Sie, wo Sie die Sonderfunktionen anwenden können und wie Sie die Sonderfunktions-OBs aufrufen und parametrieren müssen. Ferner erfahren Sie, wie Sie Fehler bei der Bearbeitung einer Sonderfunktion erkennen und per Programm bearbeiten können.

Kapitel 7 beschreibt, wofür Sie den Datenbaustein DX 0 einsetzen können und wie er aufgebaut ist. Sie werden informiert, welche Bedeutung die verschiedenen DX-0-Parameter haben. Sie lernen an Hand von Beispielen, wie Sie einen Datenbaustein DX 0 erstellen bzw. über eine Maske parametrieren können

Kapitel 8 dient als Nachschlageteil für Systemkenner. Es gibt Auskunft über die Speicherorganisation der CPU 928B und über einige Systemdatenwörter, die für den Anwender abrufbare Informationen enthalten.

Kapitel 9 wendet sich ebenfalls an Systemkenner: Diese können dort nachlesen, wie sie Daten in bestimmten Speicherbereichen über Absolutadressen ansprechen können.

Kapitel 10 gibt Ihnen zunächst einige Hinweise, wann Sie den Mehrprozessorbetrieb anwenden können und welcher Datenaustausch zwischen den CPUs und CPs dabei möglich ist. Es gibt Ihnen Informationen darüber, was Sie als Programmierer für den Mehrprozessorbetrieb tun und beachten müssen.
Schließlich erhalten Sie ausführliche Anleitungen mit Anwendungsbeispielen für den Austausch größerer Datenmengen im Mehrprozessorbetrieb (Mehrprozessorkommunikation).

Kapitel 11 informiert Sie darüber, wie Sie Ihre CPU an ein PG koppeln können und welche Hilfen Ihnen die PG-Software bietet, um Ihr STEP-5-Programm zu testen.

Kapitel 12 enthält den Anhang mit technischen Daten der auf dem AG S5-135U einsetzbaren CPUs, einige Tabellen zum Nachschlagen wichtiger Informationen zur Fehlerdiagnose sowie ein Auswertungsbeispiel des USTACKs.

Kapitel 13	gibt Ihnen einige Literaturhinweise
Kapitel 14	dient als Orientierungshilfe und enthält ein Abkürzungsverzeichnis, ein Stichwortverzeichnis sowie Verzeichnisse über alle numerierten Tabellen und Bilder.

Orientierungshilfen im Text

Um Ihnen beim Lesen oder Durchblättern schnell einen Überblick über den Inhalt einzelner Textseiten zu geben, bietet Ihnen das Handbuch neben den Überschriften für Abschnitte 2. und 3. Ordnung folgende Hilfen an:

Marginalien

Marginalien sind kursiv gedruckte Stichworte am linken Rand einer Textseite. Sie geben Auskunft über den Inhalt, der in einem oder mehreren Textabsätzen daneben behandelt wird.

Gliederungshilfen

Zur besseren Gliederung von Abschnitten 3. Ordnung werden diese Marginalien auch fett gedruckt und beziehen sich dann auf eine längere Textpassage.

Die folgenden Marginalien beziehen sich auf die aufgeführten Darstellungsarten!

Hinweise

Hinweis
Auf wichtige Informationen wird in dieser Form hingewiesen

Anleitungen

Anleitungen – oft als Handlungsfolge aufgeführt – werden in Tabellenform dargestellt z. B.:

Schritt	Aktion	Ergebnis
1	Betätigen Sie den Betriebsartenschalter von RUN nach STOP.	Die CPU befindet sich im Stoppzustand. Die STOP-LED zeigt Dauerlicht.
2	Halten Sie den Wahlschalter in Stellung OVERALL RESET fest; betätigen Sie gleichzeitig den Betriebsartenschalter von STOP nach RUN und wieder nach STOP.	URLÖSCHEN ist angefordert. Die STOP-LED blinkt schnell.

Tabellen zum Nachschlagen

Informationen, auf die Sie u. U. gezielt zugreifen wollen, werden in numerierten Tabellen in der folgenden (Beispiel) oder ähnlichen Form dargestellt und im Verzeichnis der Tabellen (siehe Kapitel 14) aufgeführt.

Tabelle 3-2 Binäre Verknüpfungsoperationen

Operation	**Operand**	**Funktion**
U		UND-Verknüpfung mit Abfrage auf Signalzustand "1"
O		ODER-Verknüpfung mit Abfrage auf Signalzustand "1"
	E 0.0 bis 127.7	eines Eingangs im PAE
	.....	.....

Beispiele

Beispiele, kurze und längere – auch mehrseitige – werden durch eine gerasterte Umrahmung hervorgehoben. Bei mehrseitigen Beispielen werden die Fortsetzseiten eindeutig gekennzeichnet.

```
Beispiel 1:   Aufruf und Parametrierung eines Funktionsbausteins mit den
              Darstellungsarten AWL und KOP/FUP in einem Programmbaustein


Darstellungsart AWL

......
```

Inhalt

Einführung 1

Inhalt von Kapitel 1

Einführung 1

Zielsetzung des Handbuchs

Dieses Handbuch soll Anwendern, die bereits Grundkenntnisse im Programmieren von Automatisierungsgeräten besitzen und die CPU 928B im Automatisierungsgerät S5-135U einsetzen wollen, spezielle Kenntnisse für das Programmieren der CPU 928B vermitteln.

Lesern, die diese Grundkenntnisse noch nicht besitzen, sei die Lektüre eines Einführungsbuches in die Programmiersprache STEP 5 /3/ oder die Teilnahme an einem Kurs in einem unserer Trainingscenter empfohlen. SIEMENS bietet Ihnen umfangreiche Schulungsmöglichkeiten zu SIMATIC S5. Nähere Informationen dazu erhalten Sie bei Ihrer SIEMENS-Geschäftsstelle.

Inhalt von Kapitel 1

Kapitel 1 informiert Sie über den Anwendungsbereich des Automatierungsgerätes S5-135U mit der CPU 928B und seine Gerätestruktur. Es erklärt die typische die Arbeitsweise einer CPU und den Aufbau eines CPU-Programms.
Sie erhalten ferner einige Vorschläge, wie Sie beim Programmieren vorgehen. können und erfahren, welche für die Programmierung wichtigen Kenndaten die CPU 928B (-3UB12) hat.
Wenn Sie bereits mit der CPU 928B (-3UB11) gearbeitet haben und wissen möchten, welche Unterschiede die CPU 928B (-3UB12) dazu hat, können Sie dies in Abschnitt 1.8 nachlesen.

1.1 Anwendungsbereich S5-135U mit CPU 928B

Einordnung in die Familie

Das Automatisierungsgerät S5-135U gehört zur Familie der speicherprogrammierbaren Steuerungen SIMATIC S5. Es ist mit der CPU 928B ein leistungsfähiges Mehrprozessorgerät zur Prozeßautomatisierung (Steuern, Melden, Überwachen, Regeln, Protokollieren). Es kann sowohl für den Aufbau einfachster Steuerungen mit binären Signalen als auch zur Lösung umfangreicher Automatisierungsaufgaben werden.

Eignung

Das AG S5-135U mit der CPU 928B eignet sich besonders gut für:

- Aufgaben, die sowohl eine sehr schnelle Bitverarbeitung als auch Wortverarbeitung erfordern, d. h. sehr schnelles Steuern und Regeln.

  Beispiele dafür sind schnelle Vorgänge im Maschinenbau (Flaschenabfüllanlage, Verpackungsmaschinen oder ähnliche Anlagen).

- Aufgaben, die eine schnelle Kommunikation mit weiteren im AG eingesetzten CPUs im Mehrprozessorbetrieb sowie mit CP-Baugruppen erfordern (z. B. bei Anbindung an Bussysteme, Visualisierungen).

- Komfortabel zu lösende Regelungsaufgaben mit der Regelungssoftware R 64 sowie deren Bedienung und Beobachtung über Standardbilder mit dem PMC-System.

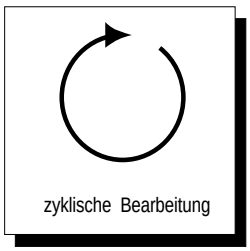
- Automatiserungsstrukturen, die eine direkte Kommunikation zu einem Leitrechner, anderen AGs oder einem Drucker über die integrierte zweite Schnittstelle bzw. SINEC-L1-Anbindung erfordern.

L e e r s e i t e

1.2 Typische Arbeitsweise einer CPU

Arbeitsweise einer CPU

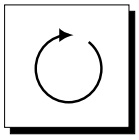
In einer CPU gibt es folgende Arbeitsweisen :

1. 

2. 

3. 

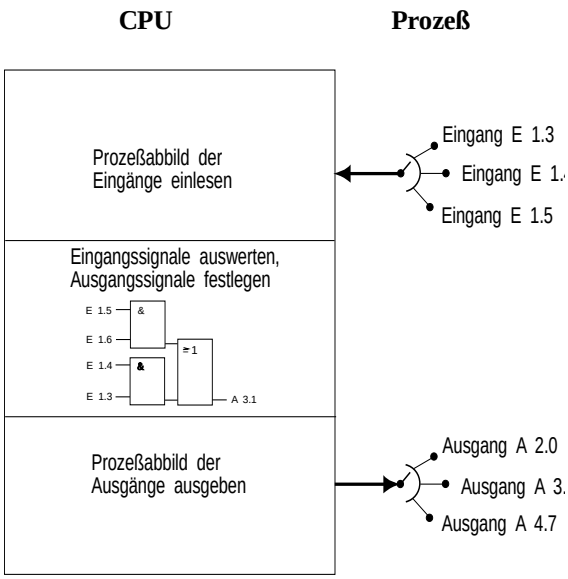
zyklische Bearbeitung



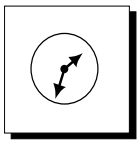
Sie stellt den Hauptanteil aller Vorgänge in der CPU. Wie ihr Name schon ausdrückt, wiederholt sich in einem endlosen Zyklus ständig die gleiche Bearbeitungsfolge.

Die zyklische Bearbeitung unterteilt sich in drei Hauptphasen :

Phase	Ablauf
1	Alle der CPU zugeordneten Eingabebaugruppen werden vom Systemprogramm abgefragt und die eingelesenen Werte im Prozeßabbild der Eingänge (PAE) zwischen-gespeichert.
2	Die im PAE enthaltenen Werte werden durch das Anwenderprogramm verarbeitet und die auszugebenden Werte in dasProzeßabbild der Ausgänge (PAA) eingetragen.
3	Die im Prozeßabbild der Ausgänge enthaltenen Werte werden vom Systemprogramm an die der CPU zugeordneten Ausgabebaugruppen ausgegeben.

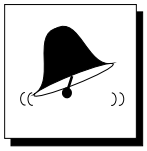


zeitgesteuerte Bearbeitung



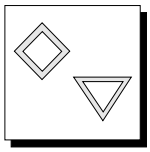
Für Prozesse, die in konstanten Zeitabschnitten Steuersignale benötigen, können Sie neben dem zyklischen Ablauf **zeitgesteuert** bestimmte Aufgaben bearbeiten z. B. zeitunkritische Überwachungsfunktionen im Sekundenraster.

alarmgesteuerte Bearbeitung



Einem Prozeßsignal, auf das besonders schnell reagiert werden muß, ordnen Sie einen **alarmgesteuerten** Bearbeitungsabschnitt zu. Sie können z. B. mit einem Prozeßalarm, der über eine eine alarmerzeugende Baugruppe ausgelöst wird, eine spezielle Bearbeitungsfolge in Ihrem Programm aktivieren.

Bearbeitung nach Priorität



Die o. g. Bearbeitungsarten werden von der CPU nach Wichtigkeitsgrad behandelt. Dieser Wichtigkeitsgrad heißt **Priorität**.

Da auf ein Zeit- oder Alarmereignis schnell reagiert werden muß, unterbricht die CPU die zyklische Bearbeitung, um ein Zeit- oder Alarmereignis zu behandeln. Die zyklische Bearbeitung hat daher die niedrigste Priorität

1.3 Die Programme in einer CPU

Das in jeder CPU vorhandene Programm unterteilt sich in

- das **Systemprogramm**

  und

- das **Anwenderprogramm**.

Systemprogramm

Das Systemprogramm organisiert alle Funktionen und Abläufe der CPU, die nicht mit einer spezifischen Steuerungsaufgabe verbunden sind (siehe Bild 1-2).

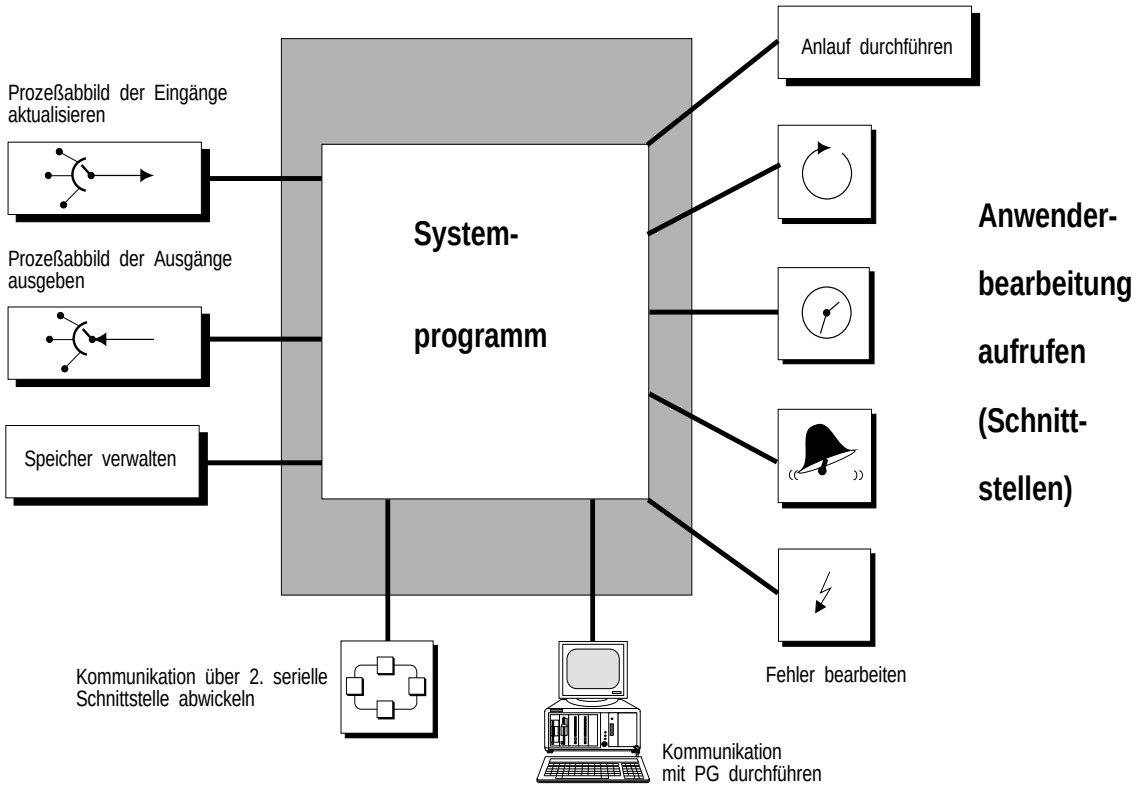


Bild 1-1 Aufgaben des Systemprogramms

Aufgaben

Zu den Aufgaben gehören : [1)]

- Neustart und Wiederanlauf,
- das Aktualisieren des Prozeßabbildes der Eingänge und die Ausgabe des Prozeßabbildes der Ausgänge,
- die Aufrufe der zyklischen, zeit- und alarmgesteuerten Programme,
- das Erkennen und Behandeln von Fehlern,
- die Speicherverwaltung,
- die Kommunikation mit dem Programmiergerät,
- die Abwicklung der Kommunikation über die 2. serielle Schnittstelle.

Anwenderschnittstellen

Als Anwender haben Sie die Möglichkeit, über spezielle Schnittstellen zum Systemprogramm das Verhalten der CPU in bestimmten Betriebs- und Fehlerfällen zu beeinflussen.

Voreinstellung des Systemverhaltens

Die nachfolgenden Kapitel außer Kapitel 7 beschreiben das **voreingestellte Systemverhalten** bei Reaktionen auf Prozeßereignisse oder Fehler. So geht z. B. entsprechend der Voreinstellung die CPU in den Stoppzustand, wenn bei einem Befehlscode-Fehler der zugeordnete Fehler-Organisationsbaustein nicht geladen ist.

Voreinstellung modifizieren

Sie können dieses Systemverhalten durch Parametrieren des Datenbausteins DX 0 modifizieren.
Wie sich **nach** einer solchen **Modifizierung** das System verhält, wird **in Kapitel 7** beschrieben!

[1)] Im Betrieb mit mehreren CPUs (Mehrprozessorbetrieb) kommen noch weitere Aufgaben hinzu.

Anwenderprogramm

Aufgaben

Das Anwenderprogramm enthält alle Funktionen, die zur Bearbeitung einer **spezifischen Steuerungsaufgabe** erforderlich sind. Diese Funktionen lassen sich bei einer groben Aufteilung direkt den Schnittstellen zuordnen, die das Systemprogramm für die unterschiedlichen Bearbeitungsarten zur Verfügung stellt:

Bearbeitungsart	Aufgabe
Neustart und Wiederanlauf	Voraussetzungen schaffen, daß die übrige Bearbeitung bei einem Neubeginn der Steuerung oder nach einem Wiederanlauf auf einem definierten Zustand aufsetzen kann (z. B. Signale mit einem bestimmten Wert vorbesetzen)
zyklische Bearbeitung	sich ständig wiederholende Signalverarbeitung (z. B. Binärsignale verknüpfen, Analogwerten einlesen und auswerten, Binärsignale für die Ausgabe festlegen, Analogwerte ausgeben).
zeitgesteuerte Bearbeitung	spezielle, zeitabhängige Bearbeitungen mit folgenden Zeitbedingungen: - schneller als der durchschnittliche Zyklus, - in einem Zeitraster, das größer ist als die durchschnittliche Zykluszeit ist, - zu einem bestimmten, einstellbaren Zeitpunkt.
alarmgesteuerte Bearbeitung	spezielle, schnelle Reaktionen auf bestimmte Prozeßsignale
Fehlerreaktion	Störungen des normalen Programmablaufs bearbeiten

Aufbau

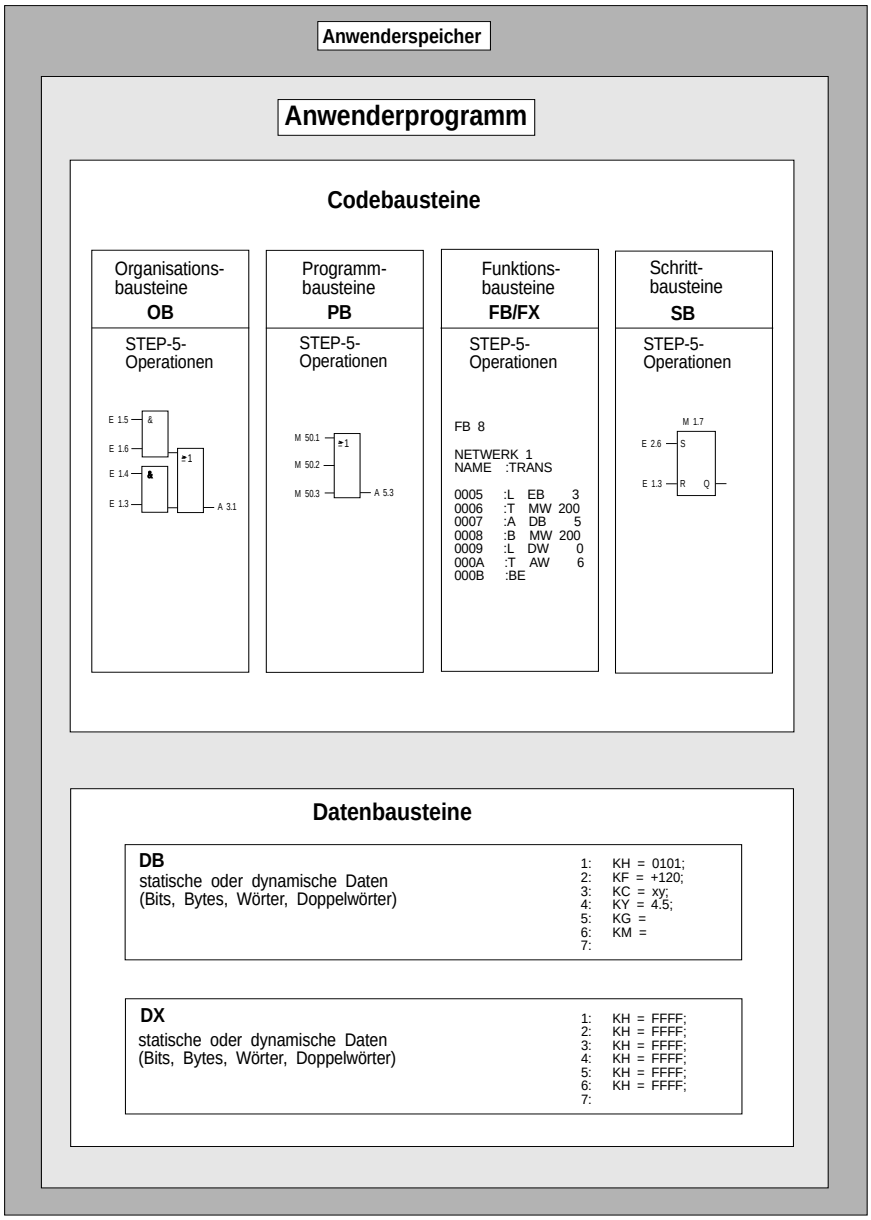


Bild 1-2 Aufbau eines STEP-5-Anwenderprogramms

Speicherung des Anwenderprogramms

Für die Ablage der Bausteine stehen auf der CPU 928B zwei Bereiche zur Verfügung:

- **Anwenderspeicher:** max. 64 K byte

  Der Anwenderspeicher befindet sich auf einem steckbaren RAM- oder EPROM-Modul und enthält Code- und Datenbausteine (falls der Anwenderspeicher ein EPROM-Modul ist, müssen Datenbausteine, deren Daten durch das Anwenderprogramm verändert werden sollen, in das DB-RAM geladen werden).

- **Datenbaustein-RAM (DB-RAM):** 46 K byte

  Das DB-RAM ist ein zusätzlicher Speicherbereich zur Aufnahme von Datenbausteinen.

Schnittstellen zum Systemprogramm

Als Schnittstellen zum Systemsystemprogramm stehen für die speziellen Bearbeitungsarten **Organisations**bausteine zur Verfügung.

1.4 Welche Operanden stehen dem Anwenderprogramm zur Verfügung?

Die CPU 928B stellt Ihnen für das Programmieren folgende Operandenbereiche zur Verfügung:

- Prozeßabbild und Peripherie
- Merker (M-Merker und S-Merker)
- Zeiten/Zähler
- Datenbausteine

Prozeßabbild der Ein- und Ausgänge PAE/PAA

Kennzeichen	Größe
Auf das Prozeßabbild kann das Anwenderprogramm sehr schnell zugreifen und zwar auf folgende Datentypen: - Einzelbits, - Bytes, - Wörter, - Doppelwörter.	je 128 byte für Ein- und Ausgänge

Peripheriebereich (P-Bereich)

Kennzeichen	Größe
Das Anwenderprogramm kann direkt über den S5-Bus auf die Peripheriebaugruppen zugreifen. Folgende Datentypen sind möglich: - Bytes, - Wörter.	je 256 byte für Ein- und Ausgänge

Erweiterter Peripheriebereich (Q-Bereich)

Kennzeichen	Größe
Das Anwenderprogramm kann direkt über den S5-Bus auf die Peripheriebaugruppen zugreifen. Folgende Datentypen sind möglich: - Bytes, - Wörter.	je 256 byte für Ein- und Ausgänge

M-Merker

Kennzeichen	Größe
Der Merkerbereich ist ein Speicherbereich, auf den das Anwenderprogramm mit entsprechenden Operationen sehr schnell zugreifen kann. Der Merkerbereich sollte bevorzugt für oft benötigte Arbeitsdaten verwendet werden. Auf folgende Datentypen kann zugegriffen werden: - Einzelbits, - Bytes, - Wörter, - Doppelwörter. Einzelne Merkerbytes können als **Koppelmerker** zum Datenaustausch zwischen den CPUs im Mehrprozessorbetrieb (siehe Kapitel 10) genutzt werden. Koppelmerker werden vom Systemprogramm am Zyklusende über einen Zwischenspeicher im Koordinator bzw. CP/IP aktualisiert.	2048 bit

S-Merker (erweiterter Merkerbereich)

Kennzeichen	Größe
Die CPU 928B enthält einen zusätzlichen Merkerbereich, den S-Merkerbereich. Hierauf kann das Anwenderprogramm wie auf die M-Merker sehr schnell zugreifen. S-Merker lassen sich jedoch **nicht** als **Aktualoperanden** bei Funktionsbaustein-Aufrufen und auch nicht als **Koppelmerker** für Datenaustausch zwischen den CPUs benutzen. Voraussetzung zu ihrer Benutzung ist die PG-Systemsoftware "S5-DOS" ab Version 3.0 oder "S5-DOS/MT" ab Version 1.0.	8192 bit

Zeiten T

Kennzeichen	Größe
Zeitzellen werden vom Anwenderprogramm mit einem Zeitwert zwischen 10ms und 9990s geladen und durch eine Startoperation von diesem Vorgabewert aus so lange im vorgewählten Zeitraster dekrementiert, bis sie den Wert Null erreicht haben.	256 Zeitzellen

Zähler Z

Kennzeichen	Größe
Zählzellen werden vom Anwenderprogramm mit einem Anfangswert (max. 999) geladen und hinauf- bzw. heruntergezählt.	256 Zähler

Datenwörter im aktuellen Datenbaustein

Kennzeichen	Größe
Ein Datenbaustein enthält Konstanten und/oder Variablen im Byte-, Wort oder Doppelwortformat. Mit STEP-5-Operationen können Sie immer auf den "aktuellen" Datenbaustein zugreifen (siehe Abschnitt 2.4.2) Auf folgende Datentypen kann zugegriffen werden: - Einzelbits, - Bytes, - Wörter, - Doppelwörter.	256 Wörter [1)]

[1)] bei Datenbausteinen, die größer als 256 Wörter lang sind, können Sie auf die Datenwörter mit der Nummer > 255 nur mit Operationen für absolute Speicherzugriffe (siehe Kapitel 9) zugreifen.

1.5 Zugriff auf Operanden- und Speicherbereiche

Für den Zugriff auf diese Operandenbereiche und den gesamten Speicher verwenden die STEP-5-Befehle zwei unterschiedliche Mechanismen:

Relative Adressierung

Der überwiegende Teil der STEP-5-Befehle adressiert eine Speicherzelle relativ zum Beginn eines Operandenbereiches. Wenn ausschließlich mit diesen Befehlen gearbeitet wird, sind Code- und Datenbereiche des Anwenderprogramms gegen ungewolltes Überschreiben geschützt. Zugleich ist das Anwenderprogramm unabhängig von der benutzten CPU, sofern die CPU über einen entsprechenden Operandenbereich verfügt.

Absolute Adressierung

Einige STEP-5-Befehle arbeiten mit absoluter Adressierung. Mit diesen Befehlen kann auf den gesamten Speicherbereich zugegriffen werden. Sie lassen sich nur in Funktionsbausteinen verwenden und sollten wegen der Gefahr einer Datenzerstörung nur mit großer Vorsicht benützt werden. Diese Befehle sind abhängig von der verwendeten CPU. Zwischen CPU 928 und CPU 928B besteht hier jedoch kein Unterschied.

Aktueller Datenbaustein

Datenbausteine werden vom Systemprogramm in den Anwenderspeicher oder das DB-RAM geladen. Ihre Lage ist abhängig vom jeweils verfügbaren Speicherplatz. Die Länge der einzelnen Datenbausteine kann unterschiedlich sein und wird beim Programmieren eines Datenbausteins festgelegt.

Der aktuelle Datenbaustein ist der Datenbaustein, dessen Anfangsadresse und Länge in speziellen Registern eingetragen ist. Dieser Eintrag erfolgt über eine spezielle STEP-5-Operation zum Aufrufen oder "Aufschlagen" (wie eine Buchseite) eines Datenbausteines. Das Anwenderprogramm kann - falls keine Befehle mit absoluter Adressierung verwendet werden - ausschließlich auf den aktuellen Datenbaustein zugreifen. Folgende Datentypen sind möglich: Einzelbits, Bytes, Wörter und Doppelwörter.

1.6 Wie können Sie beim Programmieren vorgehen?

Wenn Sie ein erfahrener Anwender sind, haben Sie bestimmt schon Ihre eigene Methode der Programmerstellung gefunden; Sie brauchen dann diesen Abschnitt nicht zu lesen.

Weniger erfahrenen Lesern seien folgende Tips gegeben, wie sie beim Entwerfen, Programmieren, Testen und Inbetriebnehmen ihres STEP-5-Programms vorgehen können.

Realisierungsabschnitte

Die Realisierung eines STEP-5-Steuerprogramm läßt sich in drei Abschnitte einteilen:

Abschnitt	Tätigkeit
1	technologische Aufgabenstellung festlegen
2	Programm entwerfen
3	Programm erstellen, testen und in Betrieb nehmen

rekursives Verfahren

In der Praxis wird sich erweisen, daß Sie einzelne Schritte wiederholen müssen ("Rekursives Verfahren"), z. B. wenn Sie bei der Verfeinerung der Aufgabenstellung feststellen, daß noch mehr Signale benötigt werden.

Schritt 1

Technologische Aufgabenstellung festlegen:

Schritt	Tätigkeit
1	Erstellen Sie ein grobes Blockdiagramm über die Steuerungsaufgaben Ihres Prozesses.
2	Erstellen Sie eine Liste der für die Aufgabe benötigten Ein- und Ausgangssignale.
3	Verfeinern Sie das Blockdiagramm, indem Sie den einzelnen Blöcken die Signale und evtl. notwendige Zeitbedingungen und/oder Zählerstände zuordnen.

Schritt 2

Programm entwerfen:

Schritt	Tätigkeit
1	Entwerfen Sie mit Hilfe des verfeinerten Blockdiagramms die notwendigen Bearbeitungsarten Ihres Programms (zyklische Bearbeitung, zeitgesteuerte Bearbeitung usw.) und benennen Sie die dazu verwendeten OBs.
2	Teilen Sie die Bearbeitungsarten in technologische und/oder funktionale Blöcke ein.
3	Prüfen Sie, ob Sie die Blöcke einem Programm- oder Funktionsbaustein zuordnen können, und benennen Sie die zu verwendenden Bausteine (PB x, FB y usw.)
4	Klären Sie, welche Zeiten, Zähler und Daten bzw. Ergebnisspeicher Sie benötigen.
5	Legen Sie die Aufgaben für jeden vorgesehenen Codebaustein und die Daten für evtl. benötigte Merker und Datenbausteine fest. Zeichnen Sie für die Codebausteine Ablaufdiagramme.

Hinweise zum Umfang der zyklischen Bearbeitung

Beachten Sie bei der Festlegung der Bearbeitungsarten folgende Randbedingungen.

- Der Zyklus muß ausreichend schnell ablaufen. Die Prozeßzustände dürfen sich nicht schneller ändern, als die CPU darauf reagieren kann. Ansonsten könnte der Prozeß außer Kontrolle geraten.

- Als maximale Reaktionszeit müssen Sie die doppelte Zykluszeit berücksichtigten.
  Die Zykluszeit wird bestimmt durch die zyklischen Bearbeitung des Systemprogramms und durch Art und Umfang des Anwenderprogramms. Sie ist oftmals nicht konstant, da das zyklische Anwenderprogramm u. U. mehrfach unterbrochen wird durch aufgerufene zeit- und alarmgesteuerte Programme.

Schritt 3

Programm erstellen, testen und in Betrieb nehmen:

Schritt	Tätigkeit
1	Legen Sie die Darstellungsart für die Codebausteine fest (KOP, FUP oder AWL, lesen Sie dazu bitte in Kapitel 2). Beachten Sie, daß Funktionsbausteine nur in der Darstellungsart AWL erstellt werden können.
2	Programmieren Sie alle Code- und Datenbausteine (die Hantierung dazu entnehmen Sie bitte Ihrem STEP-5-Handbuch).
3	Nehmen Sie die Bausteine nacheinander in Betrieb (dazu müssen Sie für jeden Teilschritt u. U. einen anderen OB 1 programmieren, der die betreffenden Codebausteine aufruft): 1a: Laden Sie den/die Baustein(e) 1b: Testen Sie den/die Baustein(e) (Die erforderlichen Hantierungen entnehmen Sie bitte Ihrem PG-Handbuch und dem Kapitel 11.)
4	Wenn Sie sicher sind, daß alle Codebausteine richtig ablaufen und alle Daten richtig berechnet und abgespeichert werden können Sie Ihr gesamtes Progamm in Betrieb nehmen.

Hinweis zur Teststrategie

Zu welchem Zeitpunkt Sie Ihr Programm das erste Mal im "heißen" Prozeßbetrieb laufen lassen, d. h. mit echten Ein- und vor allem Ausgangssignalen, müssen Sie selbst bzw. ein Team von Fachleuten entscheiden.
Je komplexer der Prozeß ist und je schwerer das Sicherheitsrisiko wiegt, um so größer muß die Sorgfalt bei der Inbetriebnahme sein.

1.7 Programmierwerkzeuge

Einsetzbare PGs

Für die Erstellung Ihres Anwenderprogramms stehen Ihnen die Programmiergeräte PG 685, PG 710, PG 730, PG 750 und PG 770 zur Verfügung. Leistung und Eigenschaften dieser Geräte können Sie dem Katalog ST 59 /9/ entnehmen.

Hinweis

- Damit Sie auch ein PG 615 oder ein OP 3xx einsetzen können, müssen Sie im Systemdatum BS 29 (siehe Kapitel 8) die CPU-Kennung für **CPU 922 (0010H)** eintragen. In diesem Fall dürfen Sie keine S-Merker benutzen.
  Wenn Sie die Kennung nicht ändern, treten fehlerhafte Anzeigen z. B. bei AUSGABE USTACK auf oder Testhilfen sind teilweise nicht möglich.
- Bei allen PGs arbeitet die **Testfunktion STATUS** uneingeschränkt nur bei Zykluszeiten ≤ 2,5 s. Im Paralellbetrieb von 2 PG-Schnittstellen (siehe Kapitel 11) halbiert sich dieser Wert.

Einsetzbare Software

Anwenderprogramme für SIMATIC-S5-Automatisierungsgeräte können Sie erstellen

- in der Programmiersprache **STEP 5** in den Darstellungsarten KOP/FUP/AWL:

  Dazu benutzen Sie das Programmierpaket STEP 5 zusammen mit der Systemsoftware STEP 5/ST bzw. STEP 5/MT (Beschreibung siehe /3/ – Literaturhinweise) ,

  oder

- in einer höheren Programmiersprache:

  Wenn Sie gewohnt sind, Programme in einer höheren Programmiersprache zu schreiben, können Sie bei der CPU 928B Ihr STEP-5-Programm auch in folgenden Sprachen formulieren:

  - **SCL** (siehe /12/ – Literaturhinweise, der SCL-Compiler ist in der PG-Software "S5-DOS/MT" ab Version 6 enthalten.)

Programme für Ablaufsteuerungen können Sie auch in einer grafischen Darstellung mit dem Programmierpaket **GRAPH 5** erstellen (Beschreibung siehe /4/ – Literaturhinweise).

Je nach Aufgabenstellung können Sie in Ihr Anwenderprogramm auch fertige Standard-Funktionsbausteine einbeziehen, deren Leistung und Eigenschaften im Katalog ST 57 /11/ aufgeführt sind.

1.8 Was ist neu bei der CPU 928B (-3UB12)?

Die CPU 928B (-3UB12) bietet Ihnen gegenüber der CPU 928B (-3UB11) folgende neue Funktionen.

zusätzliche Anlaufart: NEUSTART MIT GEDÄCHTNIS [1)]

Neben den bisherigen Anlaufarten (MANUELLER/AUTOMATISCHER NEUSTART, MANUELLER/AUTOMATISCHER WIEDERANLAUF) können Sie folgende weitere Anlaufarten benutzen:

- MANUELLER NEUSTART MIT GEDÄCHTNIS,
- AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS.

Sie können die neuen Anlaufarten durch DX-0-Parametrierung einstellen.

Verzögerungsalarm

Neben den bekannten Weckalarmen wird zusätzlich ein **Verzögerungsalarm** durch den neuen Organisationsbaustein **OB 6** bearbeitet.

Der Verzögerungsalarm hat eine zeitliche Auflösung von 1 ms.

Sie parametrieren die gewünschte Verzögerungszeit mit dem neuen Organisationsbaustein **OB 153**.

Alternatives Laden von Datenbausteinen [1)]

Sie haben die Möglichkeit, Datenbausteine mit dem PG statt in den Anwenderspeicher zuerst in das DB-RAM zu laden. Die Auswahl des Lademodus wird über Bit 0 im Systemdatenwort BS 144 gesteuert.

SINEC L1 über die 2. serielle Schnittstelle [1)]

Die Kommunikation über die 2. serielle Schnittstelle wurde Anschluß an das SINEC-L1-Bussystem (mit neuem L1-Schnittstellenmodul) erweitert:

- Einsatz als Slave bei

  - Normalverkehr,
  - Querverkehr,
  - Interrupt-Verkehr,
  - Broadcast;

- Einsatz als Master bei Punkt-zu-Punkt-Verbindungen.

[1)] nachrüstbar für CPU 928B (-3UB11)

Anwenderprogramm 2

Inhalt von Kapitel 2

Anwenderprogramm 2

Im nachfolgenden Kapitel erfaheren Sie, aus welchen Komponenten sich ein STEP-5-Anwenderprogramm für die CPU 928B zusammensetzt und wie es strukturiert werden kann.

2.1 Programmiersprache STEP 5

Mit der Programmiersprache STEP 5 setzen Sie die Ihnen vorliegenden Automatisierungs-aufgaben in Programme um, die in den SIMATIC-S5-Automatisierungsgeräten ablaufen. In STEP 5 können Sie sowohl einfache binäre Funktionen als auch komplexe digitale Funktionen und arithmetische Operationen einschließlich Gleitpunktarithmetik programmieren.

Befehlsumfang

Der **Befehlsumfang** der Programmiersprache STEP 5 gliedert sich in

Grundoperationen:

- in allen Code-Bausteinen anwendbar,
- Darstellungsarten Kontaktplan (KOP), Funktionsplan (FUP), Anweisungsliste (AWL).

Ergänzende Operationen und Systemoperationen:

- nur in Funktionsbausteinen anwendbar,
- Darstellungsart nur Anweisungsliste (AWL),
- Systemoperationen: nur für Anwender mit sehr guten Systemkenntnissen.

2.1.1 Darstellungsarten KOP, FUP und AWL

Beim Programmieren in STEP 5 können Sie für jeden einzelnen Code-Baustein zwischen den drei Darstellungsarten Kontaktplan (KOP), Funktionsplan (FUP) und Anweisungsliste (AWL) wählen, so daß die Programmiermethode dem jeweiligen Anwendungsfall angepaßt werden kann.

Der von den Programmiergeräten (PG) erzeugte Maschinencode MC 5 ist bei den drei Darstellungsarten identisch.

Wenn Sie beim Programmieren in STEP 5 bestimmte Regeln berücksichtigen (siehe /9/), kann das PG Ihr Anwenderprogramm von einer Darstellungsart in jede andere übersetzen!

Grafische Darstellung oder Liste mit Anweisungen

Während Sie mit den Darstellungsarten FUP und KOP die Möglichkeit haben, Ihr STEP-5-Programm grafisch darzustellen, werden in der Anweisungsliste die einzelnen STEP-5-Befehle aufgelistet.

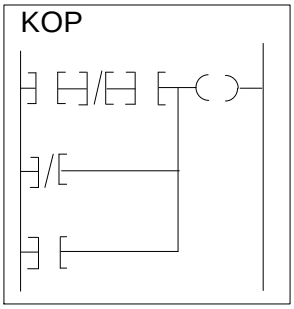
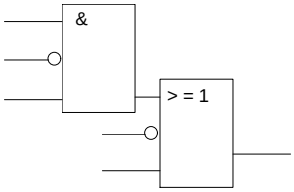
KONTAKTPLAN	ANWEISUNGSLISTE	FUNKTIONSPLAN
Programmieren mit grafischen Symbolen wie Stromlaufplan	Programmieren mit mnemotechnischen Abkürzungen der Funktionsbezeichnungen	Programmieren mit grafischen Symbolen
entspricht DIN 19239	entspricht DIN 19239	entspricht IEC 117-15 DIN 40700 DIN 40719 DIN 19239
KOP 	AWL U E UN E U E ON E O E = A	FUP 

Bild 2-1 Darstellungsarten der Programmiersprache STEP 5

Grafische Darstellung von Ablaufsteuerungen

GRAPH 5 /6/ ist eine Programmiersprache zur graphischen Darstellung von Ablaufsteuerungen. Sie ist den Darstellungsarten KOP, FUP und AWL übergeordnet. Das mit GRAPH 5 geschriebene Programm in grafischer Darstellung wird vom PG automatisch in ein STEP-5-Programm umgesetzt.

2.1.2 Strukturierte Programmierung

Mit STEP 5 kann das Anwenderprogramm strukturiert werden. Es wird dabei in einzelne, in sich abgeschlossene Programmabschnitte (Bausteine) aufgeteilt. Die Gliederung Ihres Anwenderprogramms verdeutlicht somit auf den ersten Blick die wesentlichen Programmstrukturen oder hebt programmtechnisch zusammenhängende Anlagenteile hervor.

Dieses Verfahren des **"strukturierten Programmierens"** bietet Ihnen folgende Vorteile:

- einfache und übersichtliche Programmierung auch umfangreicher Programme,
- Möglichkeit zum Standardisieren von Programmteilen,
- einfache Programmorganisation,
- leichte Änderungsmöglichkeiten,
- einfacher, abschnittsweiser Programmtest,
- einfache Inbetriebnahme.

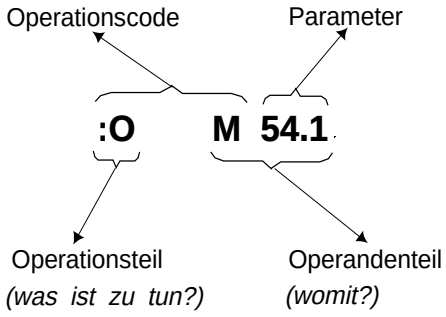
Was ist ein Baustein ?

Ein Baustein ist ein durch Funktion, Struktur oder Verwendungszweck abgegrenzter Teil des Anwenderprogramms. Man unterscheidet Bausteine, die **Anweisungen** (Code) enthalten (Organisationsbausteine, Programmbausteine, Funktionsbausteine, Schrittbausteine), und Bausteine, die **Daten** enthalten (Datenbausteine).

2.1.3 STEP-5-Operationen

Eine STEP-5-Operation ist die kleinste selbständige Einheit des Anwenderprogramms. Sie ist die Arbeitsvorschrift für die CPU. Eine STEP-5-Operation setzt sich zusammen aus einem Operationsteil und einem Operandenteil.

Beispiel



Absolute und symbolische Operanden

Den Operandenteil können Sie entweder **absolut** oder **symbolisch** (über die Zuordnungsliste) eingeben.

Beispiel für die absolute Darstellung: :U E 1.4

Beispiel für die symbolische Darstellung: :U -Motor1

Beachten Sie zur absoluten und symbolischen Programmierung die Bedienungsanleitung zu Ihrem PG.

Anwendung der STEP-5-Operationen

Der Operationsumfang von STEP 5 ermöglicht Ihnen

- binäre Werte miteinander zu verknüpfen, zu setzen oder rückzusetzen,
- Werte zu laden und zu transferieren,
- Werte miteinander zu vergleichen und arithmetisch zu bearbeiten,
- Zeit- und Zählwerte vorzugeben,
- Zahlendarstellungen umzuwandeln,
- Bausteine aufzurufen und Sprünge innerhalb eines Bausteins auszuführen

  und

- die Programmbearbeitung zu beeinflussen.

Verknüpfungsergebnis VKE

Das zentrale Bit für die Programmsteuerung ist das Verknüpfungsergebnis VKE. Es wird durch binäre Verknüpfungen gebildet sowie von einigen Operationen beeinflußt.

Der gesamte STEP-5-Operationsvorrat sowie Einzelheiten zur VKE-Bildung werden ausführlich in Abschnitt 3.5 beschrieben. Sie finden dort Programmierbeispiele zu den einzelnen STEP-5-Befehlen.

2.1.4 Zahlendarstellung

Damit die CPU Zahlenwerte miteinander verknüpfen, verändern oder vergleichen kann, müssen diese in einer binärcodierten Darstellung in die Akkumulatoren (Arbeitsregister der CPU) geladen werden.

Abhängig von der auszuführenden Operation sind in STEP 5 folgende Zahlendarstellungen zulässig:

Dualzahlen:	16-bit-Festpunktzahlen
	32-bit-Festpunktzahlen
	Gleitpunktzahlen mit 16- oder 24-bit-Mantisse
Dezimalzahlen:	BCD-codierte Zahlen (Vorzeichen und 3 Dezimalstellen)

Zahleneingabe am PG

Bei der Ein- und Ausgabe von Zahlenwerten stellen Sie am Programmiergerät das Datenformat (z. B. KF für Festpunkt) ein, in dem Sie den Zahlenwert eingeben bzw. angezeigt haben möchten. Auf diese Weise übernimmt das PG die Umrechnung aus der intern verwendeten Zahlendarstellung in die von Ihnen gewünschte Darstellungsart.

Zulässige Operationen

Mit den 16-bit-Festpunktzahlen und Gleitpunktzahlen können Sie **alle arithmetischen Operationen** wie Vergleichen, Addieren, Subtrahieren, Multiplizieren und Dividieren ausführen.

> **Hinweis**
> BCD-codierte Zahlen dürfen nicht für arithmetische Operationen verwendet werden, da sie zu falschen Ergebnissen führen.

Mit 32-bit-Festpunktzahlen werden Vergleichsoperationen durchgeführt. Außerdem werden sie bei der Umwandlung von BCD-codierten Zahlen in Gleitpunktzahlen als Zwischenstufe benötigt. Mit den Befehlen +D und -D können sie auch für Additionen und Subtraktionen verwendet werden.

Die STEP-5-Sprache enthält **Umwandlungsoperationen**, mit denen Sie Zahlen direkt in die wichtigsten Zahlendarstellungen umwandeln können.

16- und 32-bit-Festpunktzahlen

Festpunktzahlen sind ganze, mit einem Vorzeichen versehene Dualzahlen.

Codierung der Festpunktzahlen

Sie sind in binärcodierter Darstellung 16 bit (= 1 Wort) bzw. 32 bit (= 2 Wörter) breit, wobei Bit-Nr. 15 bzw. Bit-Nr. 31 das Vorzeichen enthält:

- ’0’ = positive Zahl
- ’1’ = negative Zahl

Negative Zahlen werden im 2er-Komplement dargestellt.

PG-Eingabe

Eingabe des Datenformats für 16-bit-Festpunktzahlen am PG: KF

Eingabe des Datenformats für 32-bit-Festpunktzahlen am PG: DH

zulässiger Zahlenbereich

-32768 bis +32767 (16 bit)

-2147483648 bis +2147483647 (32 bit)

Anwendung der Festpunktzahlen

Festpunktzahlen werden bei einfachen Rechenaufgaben und beim Vergleich von Zahlenwerten verwendet. Da Festpunktzahlen immer ganze Zahlen sind, beachten Sie bitte, daß das Ergebnis einer Division von zwei Festpunktzahlen ebenfalls eine Festpunktzahl ohne Dezimalstellen ist.

Gleitpunktzahlen

Gleitpunktzahlen sind positive und negative gebrochene Zahlen. Sie belegen immer ein Doppelwort (32 bit). Eine Gleitpunktzahl wird als Exponentialzahl dargestellt. Die Mantisse ist 16 oder 24 bit, der Exponent 8 bit lang.

Die CPU 928B rechnet – sofern nichts anderes eingestellt wurde – beim Addieren, Subtrahieren, Multiplizieren und Dividieren mit einer 16 bit breiten Mantisse (Bit 8 bis 23). Die niederwertigen (rechts stehenden) Bit 0 bis 7 haben dabei immer den Wert ’0’!

Wünschen Sie bei Gleitpunktrechnungen eine höhere Genauigkeit (und können Sie dafür kleine Laufzeitverluste in Kauf nehmen), so programmieren Sie im DX 0 die Einstellung "Gleitpunktarithmetik mit 24-bit-Mantisse" (siehe Kapitel 7).

Der Exponent gibt die Größenordnung der Gleitpunktzahl an. Am Vorzeichen des Exponenten erkennen Sie, ob der Betrag der Gleitpunktzahl größer oder kleiner als 0,1 ist.

Anwendung der Gleitpunktzahlen

Verwenden Sie Gleitpunktzahlen für die Lösung umfangreicherer Rechenaufgaben, insbesondere bei Multiplikationen und Divisionen, und dann, wenn Sie mit sehr großen oder sehr kleinen Zahlen arbeiten!

Genauigkeit

Die Mantisse gibt die Genauigkeit der Gleitpunktzahl an:

- Genauigkeit bei einer 24-bit-Mantisse:

  $\mathbf{2^{-24} = 0{,}000000059604}$ (entspricht 7 Nachkommastellen)

- Genauigkeit bei einer 16-bit-Mantisse:

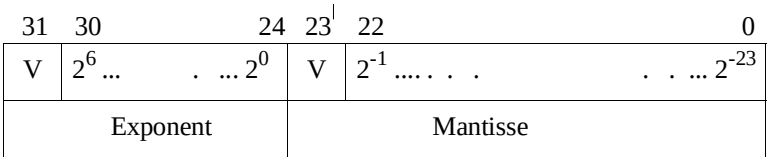
  $\mathbf{2^{-16} = 0{,}000015258}$ (entspricht 4 Nachkommastellen)

Ist das Vorzeichen der Mantisse ’0’, ist die Zahl positiv; bei Vorzeichen ’1’ ist es eine negative Zahl in 2er-Komplement-Darstellung.

Der **Gleitpunktwert ’0’** wird als dualer Wert **80000000H** (32 bit, siehe "Codierung der Gleitpunktzahlen") dargestellt.

Codierung der Gleitpunktzahlen

Codierung einer Gleitpunktzahl:



Angabe des Datenformats für Gleitpunktzahlen am PG: KG

zulässiger Zahlenbereich

$\pm 0{,}1469368 \times 10^{-38}$ bis $\pm 0{,}1701412 \times 10^{39}$

Ein-/Ausgabe am PG

a) im Code-Baustein:

Es soll die Zahl Z = 12,34567 als Gleitpunktkonstante geladen werden.

Eingabe:

:LKG1234567+2

Ausgabe durch PG nach Übernahme der Zeile:

:L KG + 1234567 + 02

Mantisse mit Vorzeichen — Exponent (Basis 10) mit Vorzeichen

Wert der eingebenen Zahl: $+0{,}1234567 \times 10^{+2} = 12{,}34567$

b) im Daten-Baustein:

Es soll die Zahl Z = - 0,005 als Gleitpunktkonstante definiert werden.

Eingabe:

6: KG = - 5 - 2

Ausgabe durch PG nach Übernahme der Zeile:

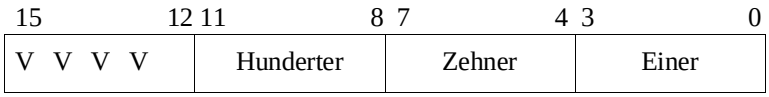
6: KG =- 5000000 - 02

Mantisse mit Vorzeichen — Exponent (Basis 10) mit Vorzeichen

Wert der eingebenen Zahl: $-0{,}5 \times 10^{-2} = 0{,}005$

BCD-codierte Zahl

Dezimalzahlen werden als BCD-Zahlen dargestellt. Mit Vorzeichen und 3 Ziffern belegen sie im Akkumulator 16 bit (1 Wort):



Die einzelnen Ziffern sind positive 4-bit-Dualzahlen zwischen 0000 und 1001 (0 und 9 dezimal).

Die linken Bit sind für das Vorzeichen reserviert.

Vorzeichen bei einer positiven Zahl:	0000
Vorzeichen bei einer negativen Zahl:	1111

zulässiger Zahlenbereich

-999 bis +999

2.1.5 STEP-5-Bausteine und deren Ablage im Speicher

Kennzeichnung

Ein Baustein ist gekennzeichnet durch:

- den Bausteintyp (OB, PB, SB, FB, FX, DB, DX)

  und

- die Bausteinnummer (Zahl zwischen 0 und 255).

Bausteintypen

Die Programmiersprache STEP 5 unterscheidet folgende Bausteintypen:

Organisationsbausteine (OB)

Die Organisationsbausteine sind die Schnittstelle zwischen dem Systemprogramm und dem Anwenderprogramm. Sie können in zwei Gruppen unterteilt werden:

Mit den OB 1 bis 39 können Sie die Programmbearbeitung, das Anlaufverhalten der CPU und das Verhalten im Fehlerfall steuern, indem Sie diese Bausteine entsprechend den Ihnen vorliegenden Automatisierungsaufgaben programmieren. Diese OB werden vom Systemprogramm aufgerufen.

Die OB 40 bis 255 enthalten Sonderfunktionen des Systemprogramms. Sie werden bei Bedarf vom Anwenderprogramm aufgerufen.

Programmbausteine (PB)

Programmbausteine werden zur Strukturierung des Anwenderprogramms verwendet und enthalten die nach technologischen oder funktionellen Gesichtspunkten gegliederten Teilprogramme. Die PB bilden den Kern des Anwenderprogramms.

Schrittbausteine (SB)

Schrittbausteine waren ursprünglich spezielle Programmbausteine zur "schritt"weisen Bearbeitung von Ablaufketten. Ablaufketten können jedoch inzwischen über GRAPH 5 /6/ programmiert werden. Daher haben Schrittbausteine in STEP 5 nicht mehr ihre ursprüngliche Bedeutung.
Schrittbausteine sind jetzt eine zahlenmäßige Erweiterung der Programmbausteine. Sie können wie diese eingesetzt werden.

Funktionsbausteine (FB/FX)

Funktionsbausteine dienen zum Programmieren von häufig wiederkehrenden oder auch komplexen Funktionen (z.B. digitale Funktionen, Ablaufsteuerungen, Regelungen, Meldefunktionen).

Ein Funktionsbaustein kann von übergeordneten Bausteinen mehrfach aufgerufen werden und bei jedem Aufruf mit neuen Operanden versorgt ("parametriert") werden.
Durch die Verwendung der Bausteinart FX wird die Zahl der maximal möglichen Funktionsbausteine von 256 auf 512 erhöht.

Datenbausteine (DB/DX)

In Datenbausteinen stehen die (festen oder veränderbaren) Daten, mit denen das Anwenderprogramm arbeitet. Diese Bausteinart enthält keine STEP-5-Anweisungen und unterscheidet sich in ihrer Funktion grundsätzlich von den übrigen Bausteinen. Durch die Verwendung der Bausteinart DX wird die Zahl der maximal möglichen Datenbausteine verdoppelt.

Aufbau der Bausteine

Alle Bausteintypen bestehen aus

- einem Bausteinkopf

  und

- einem Bausteinrumpf.

Bausteinkopf

Der **Bausteinkopf** hat immer eine Länge von 5 Wörtern und enthält Informationen für die Bausteinverwaltung im PG und Daten für das Systemprogramm.

Bausteinrumpf

Im **Bausteinrumpf** sind – abhängig vom Bausteintyp – enthalten:

- STEP-5-Befehle (bei OB, PB, SB, FB, FX),
- variable oder konstante Daten (bei DB, DX)

  und

- Formaloperandenliste (bei FB, FX).

Bausteinvorkopf

Zu den Bausteinen vom Typ DB, DX, FB und FX erzeugt das Programmiergerät zusätzlich einen **Bausteinvorkopf** (DV, DXV, FV, FXV). Diese Bausteinvorköpfe enthalten Informationen über das Datenformat (bei DB und DX) bzw. über die Sprungmarken (bei FB und FX), die nur das Programmiergerät auswerten kann. Die Bausteinvorköpfe werden deshalb nicht in den CPU-Speicher übertragen. Als Anwender haben Sie auf den Inhalt eines Bausteinvorkopfes direkt keinen Einfluß.

Maximale Länge

Ein STEP-5-Baustein darf maximal 4096 Wörter (1 Wort entspricht 16 bit) im Programmspeicher der CPU belegen.

Verfügbare Bausteine

Von den einzelnen Bausteintypen stehen Ihnen zum Programmieren zur Verfügung:

OB	1 bis 39	
FB	0 bis 255	insgesamt 512
FX	0 bis 255	
PB	0 bis 255	
SB	0 bis 255	
DB	3 bis 255	insgesamt 506
DX	3 bis 255	

Die Datenbausteine DB 0, DB 1, DB2, DX 0 , DX 1 und DX 2 enthalten Parameter. Sie sind für bestimmte Funktionen reserviert und deshalb nicht beliebig verwendbar.

Ablage der Bausteine

Alle programmierbaren Bausteine werden vom PG in der Reihenfolge ihres Transfers im Programmspeicher hinterlegt (Bild 2-2). Mit der PG-Funktion "Übertragen Bausteine B" werden zunächst die Code- und dann die Datenbausteine zum AG übertragen. Bei RAM-Betrieb wird zuerst das RAM-Modul nach Übertragung der Codebausteine mit Datenbausteinen aufgefüllt und danach werden die restlichen Datenbausteine in das interne DB-RAM geschrieben.
Die Anfangsadressen der gespeicherten Bausteine werden im Datenbaustein DB 0 hinterlegt.

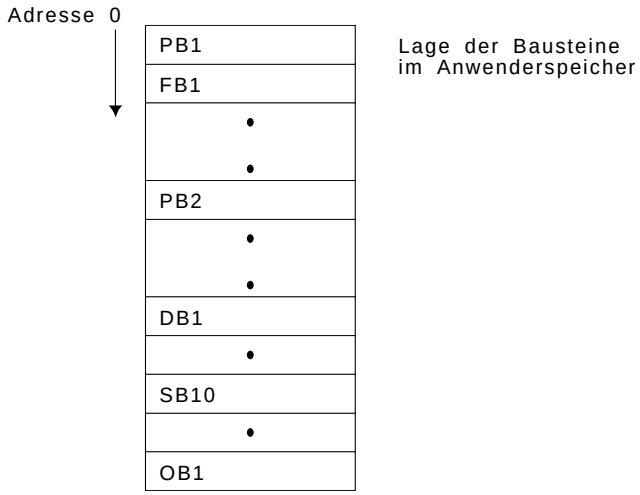


Bild 2-2 Beispiel für eine Ablage der Bausteine im Programmspeicher

Alternatives Laden (nur bei Version -3UB12)

Durch Setzen von Bit 0 des Systemdatums BS 144 können Sie Datenbausteine vorrangig (d. h. solange Platz vorhanden ist) in das interne DB-RAM laden ("alternatives Laden" – siehe Kapitel 8/BS 144). Erst wenn das DB-RAM gefüllt ist, werden Datenbausteine in das RAM-Modul übertragen.

Korrigieren und Löschen von Bausteinen

Beim **Korrigieren** von Bausteinen im "RAM-Betrieb" wird der "alte" Baustein im Speicher für ungültig erklärt und ein neuer Baustein im Speicher eingetragen.
Ebenso werden beim **Löschen** von Bausteinen die Bausteine nicht wirklich gelöscht, sondern nur für ungültig erklärt. Gelöschte und korrigierte Bausteine belegen also weiterhin Speicherplatz.

Hinweis
Mit der Online-Funktion SPEICHER KOMPRIMIEREN schaffen Sie Speicherplatz für neue Bausteine: Die Funktion optimiert die Speicherplatzbelegung, indem sie ungültig markierte Bausteine löscht und die gültigen Bausteine zusammenschiebt. Das Zusammenschieben erfolgt getrennt nach Speichermodul und internem RAM (siehe Abschnitt 11.2.2).

2.2 Programm-, Organisations- und Schrittbausteine

Programmbausteine (PB), Organisationsbausteine (OB) und Schrittbausteine (SB) unterscheiden sich hinsichtlich Programmierung und Aufruf nicht. Alle drei können Sie wahlweise in den Darstellungsarten KOP, FUP und AWL programmieren.

Programmieren

Gehen Sie beim Programmieren von Programm-, Organisations- und Schrittbausteinen folgendermaßen vor:

Schritt	Aktion
1	Geben Sie zuerst den Bausteintyp, dann die Nummer des Bausteins an, den Sie programmieren wollen. Folgende Nummern stehen Ihnen jeweils zur Verfügung: - Programmbausteine 0 bis 255 - Organisationsbausteine 1 bis 39 - Schrittbausteine 0 bis 255,
2	Geben Sie Ihr Anwenderprogramm in STEP 5 ein. Beim Programmieren von PB, OB und SB dürfen Sie nur die STEP-5-**Grund**operationen verwenden! Ein STEP-5-Baustein sollte immer einen in sich abgeschlossenen Programmabschnitt enthalten. Logische Verknüpfungen müssen innerhalb eines Bausteins abgeschlossen sein.
3	Schließen Sie die Programmeingabe mit der Anweisung "BE" (Bausteinende) ab.

Aufrufen

Bausteine – außer OB 1 bis OB 39 – müssen zum Bearbeiten aufgerufen werden. Dies geschieht durch spezielle STEP-5-Operationen, die Bausteinaufrufe.
Diese Bausteinaufrufe können innerhalb eines Organisations-, Programm-, Funktions- oder Schrittbausteins programmiert werden. Sie sind vergleichbar mit Sprüngen in ein Unterprogramm. Jeder Sprung verursacht einen Bausteinwechsel. Die Rücksprungadresse im aufrufenden Baustein wird vom System zwischengespeichert.

Aufrufe können sowohl unbedingt als auch bedingt ausgeführt werden:

Unbedingter Aufruf

Die Anweisung SPA gehört zu den unbedingten Operationen. Sie hat selbst keinen Einfluß auf das VKE. Dieses wird beim Sprung in den neuen Baustein mitgenommen. Dort kann es zwar ausgewertet, jedoch nicht mehr weiter verknüpft werden.

Der angesprochene Baustein wird **unabhängig** vom Verknüpfungsergebnis (VKE – siehe Abschnitt 3.4) bearbeitet.

Beispiel: **SPA** PB 100

Bedingter Aufruf

Die Anweisung SPB gehört zu den bedingten Operationen, d. h. der angesprochene Baustein wird nur bearbeitet, wenn das Verknüpfungsergebnis **VKE = 1** ist. Bei VKE = 0 wird die Sprunganweisung nicht ausgeführt.

Beispiel: **SPB** PB 100

Hinweis
Nach Ausführen der bedingten Sprungoperation ist das VKE auf **'1'** gesetzt unabhängig davon, ob der Sprung in den aufgerufenen Baustein ausgeführt wird oder nicht.

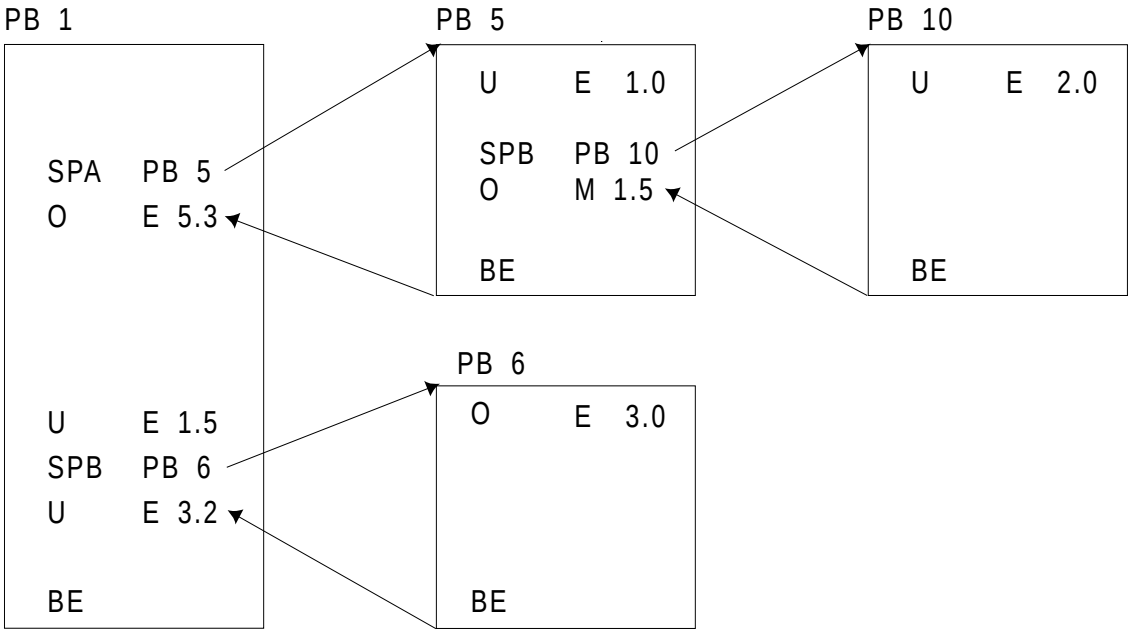


Bild 2-3 Bausteinaufrufe, die die Bearbeitung eines Programmbausteins freigeben

Was die BE-Anweisung bewirkt

Nach der Anweisung BE (Bausteinende) setzt die CPU das Anwenderprogramm in **dem** Baustein fort, in dem der **Bausteinaufruf** programmiert wurde: Die Programmbearbeitung wird bei der auf den Bausteinaufruf folgenden STEP-5-Anweisung fortgesetzt.

Die Anweisung BE wird unabhängig vom Verknüpfungsergebnis bearbeitet. Nach BE kann das Verknüpfungsergebnis nicht mehr weiter verknüpft werden. Das bei Ausführung des BE-Befehls vorhandene Verknüpfungs-/Rechenergebnis wird jedoch an den aufrufenden Baustein übergeben und kann dort ausgewertet werden. Bei der Rückkehr aus dem aufgerufenen Baustein werden die Inhalte von AKKU 1, AKKU 2, AKKU 3 und AKKU 4, die Anzeigen ANZ 0 und ANZ 1 und das Verknüpfungsergebnis VKE nicht verändert (nähere Erläuterungen zu den AKKUs, ANZ 0/ANZ 1 und VKE siehe Abschnitt 3.5).

2.2.1 Organisationsbausteine für Anwenderschnittstellen

Die Schnittstelle zwischen dem Systemprogramm und dem Anwenderprogramm sind die Organisationsbausteine. Die Organisationsbausteine OB 1 bis 39 sind Teile des Anwenderprogramms, die Sie genauso wie Programmbausteine programmieren. Durch Programmierung dieser OB können Sie das Verhalten der CPU beim Anlauf, während der Programmbearbeitung und im Fehlerfall beeinflussen. Die Organisationsbausteine sind wirksam, sobald sie in den AG-Speicher geladen werden. **Dies ist auch im laufenden Betrieb möglich.**

Nachdem das Systemprogramm den betreffenden Organisationsbaustein aufgerufen hat, wird das darin enthaltene Anwenderprogramm bearbeitet.

Hinweis
Die Organisationsbausteine OB 1 bis OB 39 für Anwenderschnittstellen werden vom Anwender programmiert und vom Systemprogramm als Reaktion auf bestimmte Ereignisse automatisch aufgerufen!

Zu **Testzwecken** können diese Organisationsbausteine auch vom Anwenderprogramm aufgerufen werden (SPA/SPB OB xxx). Es ist jedoch nicht möglich, z.B. durch Aufruf von OB 20 einen NEUSTART auszulösen!

Die nachfolgenden Tabellen geben Ihnen eine Übersicht über die Anwendernahtstellen (OBs).

Tabelle 2-1 Übersicht der OBs für die Programmbearbeitung

Organisationsbausteine zum Steuern der Programmbearbeitung	
Baustein	**Funktion und Aufrufkriterium**
OB 1	Organisation der zyklischen Programm-bearbeitung: Erster Aufruf nach Ende einer Anlaufart, dann zyklischer Aufruf.
OB 2	Organisation der alarmgesteuerten Programm-bearbeitung; Aufruf durch Interrupt-Signal des S5-Bus (Prozeßalarm)
OB 3 bis 5	bei CPU 928B nicht vorhanden
OB 6	Verzögerungsalarm (ab Version -3UB12)
OB 7, OB 8	bei CPU 928B nicht vorhanden
OB 9	Uhrzeitgesteuerter Weckalarm
	Weckalarm mit festen Zeitrastern:
OB 10	Aufruf alle 10 ms
OB 11	Aufruf alle 20 ms
OB 12	Aufruf alle 50 ms
OB 13	Aufruf alle 100 ms
OB 14	Aufruf alle 200 ms
OB 15	Aufruf alle 500 ms
OB 16	Aufruf alle 1 s
OB 17	Aufruf alle 2 s
OB 18	Aufruf alle 5 s

Tabelle 2-2 Übersicht der OBs für den Anlauf

Organisationsbausteine zum Steuern des Anlaufverhaltens	
Baustein	**Funktion und Aufrufkriterium**
OB 20	Aufruf bei NEUSTART (manuell und automatisch)
OB 21	Aufruf bei MANUELLEM WIEDER-ANLAUF/NEUSTART MIT GEDÄCHTNIS
OB 22	Aufruf bei AUTOMATISCHEM WIEDER-ANLAUF/NEUSTART MIT GEDÄCHTNIS

Tabelle 2-3 Übersicht der OBs für die Fehlerbearbeitung

Organisationsbausteine für Reaktionen auf Geräte- oder Programmfehler [1)]	
Baustein	**Funktion und Aufrufkriterium**
OB 19	Laufzeitfehler (LZF): Aufruf eines nicht geladenen Bausteins
OB 23	Quittungsverzug (QVZ) im Anwenderprogramm (bei Direktzugriff auf Peripheriebaugruppen oder andere S5-Bus-Adressen)
OB 24	Quittungsverzug (QVZ) beim Aktualisieren des Prozeßabbildes und bei Koppelmerkerübertragung
OB 25	Adressierfehler (ADF)
OB 26	Zykluszeitüberschreitung (ZYK)
OB 27	Befehlscodefehler (BCF): Substitutionsfehler
OB 28	STOP durch PG-Funktion/Stoppschalter/ S5-Bus [2)]
OB 29	Befehlscodefehler (BCF): Operationscode nicht zulässig
OB 30	Befehlscodefehler (BCF): Parameter nicht zulässig
OB 31	sonstige Laufzeitfehler (LZF)
OB 32	Laufzeitfehler (LZF): Lade- und Transferfehler bei Datenbausteinen
OB 33	Weckfehler (WECK-FE)

Organisationsbausteine für Reaktionen auf Geräte- oder Programmfehler [1]	
Baustein	**Funktion und Aufrufkriterium**
Fortsetzung der Tabelle 2-3:	
OB 34	Fehler bei Reglerbearbeitung (REG-FE)
OB 35	Kommunikationsfehler auf der zweiten seriellen Schnittstelle (FE-3)
OB 36 bis 39	bei CPU 928B nicht vorhanden

1) Ist im Fehlerfall der OB nicht geladen, geht die CPU in den Stoppzustand.
AUSNAHME: Bei nicht vorhandenem OB 23, OB 24 und OB 35 erfolgt keine Reaktion!

2) Der OB 28 wird vor Übergang in den Stoppzustand aufgerufen. Der Stoppzustand erfolgt immer, gleichgültig ob und wie der OB 28 programmiert ist.
AUSNAHME: Bei NETZ AUS wird der OB 28 nicht aufgerufen!

2.2.2 Sonderfunktions-Organisationsbausteine

Die folgenden Organisationsbausteine enthalten Sonderfunktionen des Systemprogramms. Sie können von Ihnen **nicht** programmiert (dies gilt für alle OB mit Nummern zwischen 40 und 255!), sondern lediglich aufgerufen werden. Sie enthalten kein STEP-5-Programm. Sonderfunktions-OB können in allen Code-Bausteinen aufgerufen werden.

Tabelle 2-4 Übersicht der Sonderfunktions-Organisationsbausteine

Integrierte Organisationsbausteine mit Sonderfunktionen	
Baustein:	**Bausteinfunktion:**
OB 110	Zugriff auf Anzeigenbyte
OB 111	AKKU 1, 2, 3 und 4 löschen
OB 112	AKKU Roll Up
OB 113	AKKU Roll Down
OB 120	"Alarme gemeinsam sperren" ein-/ausschalten
OB 121	"Weckalarme einzeln sperren" ein-/ausschalten
OB 122	"Alarme gemeinsam verzögern" ein-/ausschalten
OB 123	"Weckalarme einzeln verzögern" ein-/ausschalten
OB 150	Systemzeit stellen/lesen
OB 151	Uhrzeitgesteuerte Weckalarmzeit stellen/lesen
OB 152	Zyklusstatistik
OB 153	Zeit für Verzögerungsalarm stellen/lesen
OB 160 bis 163	Zählschleifen
OB 170	Bausteinstack (BSTACK) lesen
OB 180	variabler Datenbaustein-Zugriff
OB 181	Datenbausteine DBDX testen
OB 182	Datenbereich kopieren
OB 190, 192	Merker in Datenbaustein übertragen
OB 191, 193	Datenblöcke in Merkerbereich übertragen
OB 200, 202 bis 205	Funktionen zur Mehrprozessor-Kommunikation
OB 216 bis 218	Zugriffe auf "Kachel" (CPs und einige IPs)
OB 220	Vorzeichenerweiterung
OB 221	Zyklusüberwachungszeit einstellen
OB 222	Zyklusüberwachungszeit neu starten

Integrierte Organisationsbausteine mit Sonderfunktionen	
Baustein:	**Bausteinfunktion:**
Fortsetzung der Tabelle 2-4:	
OB 223	Anlaufarten vergleichen
OB 224	Koppelmerker blockweise übertragen
OB 226	Wort aus Systemprogramm lesen
OB 227	Quersumme des Systemprogrammspeichers lesen
OB 228	Statusinformation einer Programmbearbeitungs-ebene lesen
OB 230 bis 237	Funktionen für Standard-Funktionsbausteine (Hantierungsbausteine)
OB 240 OB 241 OB 242	Schieberegister initialisieren Schieberegister bearbeiten Schieberegister löschen
OB 250 OB 251	Regelung: PID-Algorithmus initialisieren Regelung: PID-Algorithmus bearbeiten
OB 254, 255	Datenbaustein ins DB-RAM übertragen

Die ausführliche Beschreibung dieser Sonderfunktionen finden Sie in Kapitel 6.

2.3 Funktionsbausteine

Funktionsbausteine (FB/FX) sind ebenso Teile des Anwenderprogramms wie z. B. Programmbausteine. FX-Funktionsbausteine haben den gleichen Aufbau wie FB-Funktionsbausteine und werden ebenso programmiert.
Mit Funktionsbausteinen werden häufig wiederkehrende oder sehr komplexe Funktionen realisiert. Jeder Funktionsbaustein stellt innerhalb des Anwenderprogramms eine abgeschlossene Funktion dar. Sie können Funktionsbausteine

- als Softwareprodukt von SIEMENS beziehen (Standard-Funktionsbausteine auf Diskette – siehe /5/); mit diesen Standard-Funktionsbausteinen können Sie Ihre Proramme für Steuern, Melden, Regeln und Protokollieren schnell und sicher erstellen)

  oder

- selbst programmieren.

Funktionsbausteine weisen gegenüber den Organisations-, Programm- und Schrittbausteinen vier wesentliche **Unterschiede** auf:

	OB, PB, SB	**FB/FX**
1.	**Operationsumfang**	
	nur Grundoperationen	- Grundoperationen, - Ergänzende Operationen - Systemoperationen
2.	**Darstellungsart**	
	Programmieren und Aufrufen in AWL, KOP, FUP	Programmieren nur in AWL
3.	**Namen**	
	Namensgebung nicht möglich (nur Nummer)	zusätzlich zur Nummer kann ein Name mit max. 8 Zeichen zugewiesen werden
4.	**Operanden**	
	keine	Formaloperanden (Bausteinparameter). Beim Aufruf werden den Formaloperanden Aktualoperanden zugewiesen.

2.3.1 Aufbau von Funktionsbausteinen

Der **Bausteinkopf** (5 Wörter) eines Funktionsbausteins ist gleich aufgebaut wie die Bausteinköpfe der übrigen STEP-5-Bausteine.

Der **Bausteinrumpf** hingegen unterscheidet sich in seinem Aufbau von dem der anderen Bausteinarten. Die auszuführende Funktion ist darin in Form einer Anweisungsliste in der Programmiersprache STEP 5 geschrieben. Zwischen dem Bausteinkopf und den STEP-5-Anweisungen benötigt ein Funktionsbaustein weiteren Speicherplatz für die Angabe seines Namens und für die Liste der Formaloperanden. Da diese Liste keine Anweisungen für die CPU enthält, wird sie mit einem unbedingten Sprung übersprungen, den das PG automatisch erzeugt. Diese Sprunganweisung wird bei der Ausgabe am PG nicht angezeigt! Bei einem Aufruf des Funktionsbausteins wird nur der Bausteinrumpf bearbeitet.

absolute oder symbolische Operanden

Operanden können in einem Funktionsbaustein absolut (z. B. M 2.5) oder symbolisch (z. B. -MOTOR1) eingegeben werden. Die Zuordnung der symbolischen Operanden müssen Sie zuvor in einer **Zuordnungsliste** (siehe /3/) ablegen.

So sieht ein Funktionsbaustein im AG-Speicher aus:

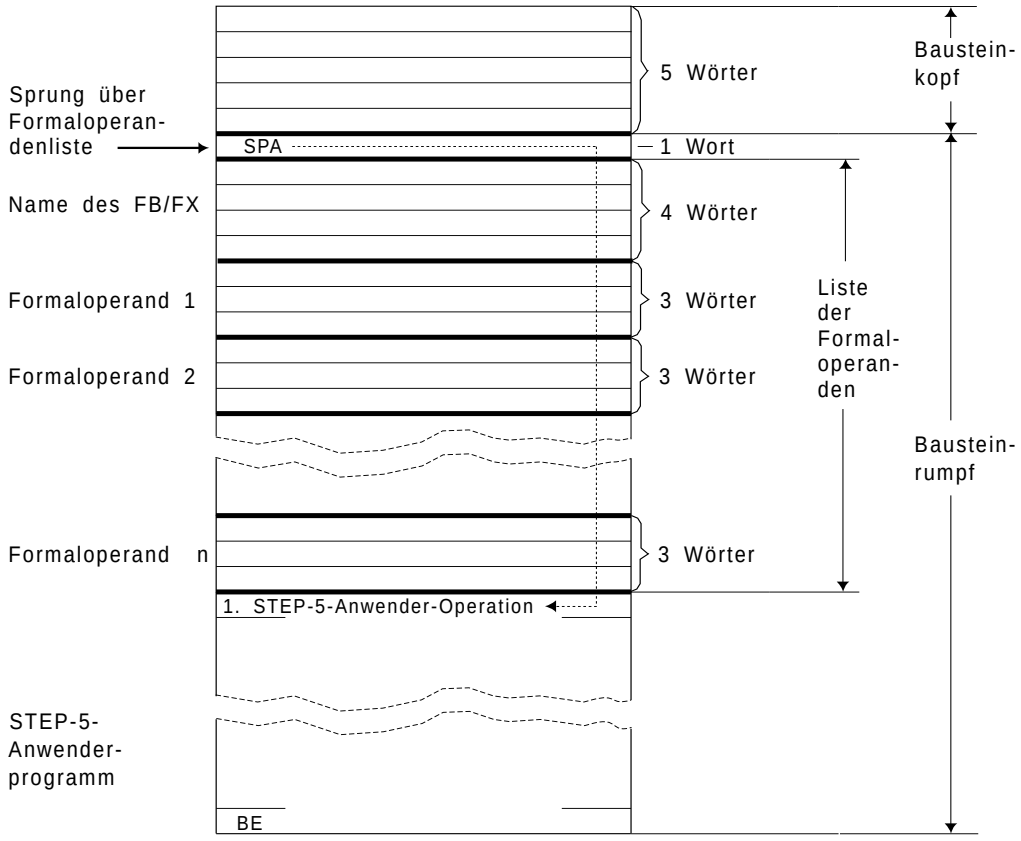


Bild 2-4 Aufbau eines Funktionsbausteins (FB/FX)

Im Speicher stehen somit alle Angaben, die das Programmiergerät benötigt, um den Funktionsbaustein beim Aufruf grafisch darstellen zu können und um die Operanden bei der Parametrierung und Programmierung des Funktionsbausteins zu überprüfen. Eine fehlerhafte Eingabe wird vom Programmiergerät abgelehnt.

Unterscheiden Sie bei der Behandlung von Funktionsbausteinen zwischen

- FB/FX **programmieren**

  und

- FB/FX **aufrufen** und anschließend **parametrieren**.

Unterschied: "Programmieren" – "Aufrufen und Parametrieren"

Beim **Programmieren** legen Sie die Funktion des Bausteins fest. Hierbei müssen Sie überlegen, welche Eingangsoperanden die Funktion benötigt und welche Ausgangsergebnisse sie an das aufrufende Programm übergeben soll. Alle Eingangsoperanden und Ausgangsergebnisse definieren Sie dann als Formaloperanden. Diese üben eine Platzhalterfunktion aus.

Beim **Aufrufen** eines Bausteins durch einen übergeordneten Baustein (OB, PB, SB, FB, FX) werden die Formaloperanden (Bausteinparameter) durch Aktualoperanden ersetzt: der Funktionsbaustein wird **parametriert.**

Vorgehen beim Programmieren

WENN...	**DANN...**
Sie einen Funktionsbaustein "direkt", d. h. ohne Formaloperanden programmieren wollen.	führen Sie dies durch wie bei Programm- oder Schrittbausteinen.
Sie in einem Funktionsbaustein Formaloperanden verwenden wollen,	gehen Sie so vor, wie auf den nächsten Seiten beschrieben. Beachten Sie dabei die erforderliche Reihenfolge: 1. den FB/FX mit den Formaloperanden programmieren und im PG (offline) oder im CPU-Speicher (online) halten, 2. den/die aufrufenden Bausteine(e) mit den Aktualoperanden programmieren.

2.3.2 Programmieren von Funktionsbausteinen

Einen Funktionsbausteins können Sie nur in der Darstellungsart **"Anweisungsliste"** programmierenn. Bei der Eingabe eines Funktionsbausteins am PG gehen Sie so vor:

Schritt	Aktion
1	Geben Sie den **Bausteintyp** (FB/FX) und die **Nummer** des Funktionsbausteins ein. Anwender-Funktionsbausteine sollten von FB 255 an fallend numeriert werden, um nicht mit den Standard-Funktionsbausteinen zu kollidieren, die von FB 1 bis FB 199 numeriert sind.
2	Geben Sie den **Namen** des Funktionsbausteins ein. Der Name kann bis zu 8 Zeichen lang sein und muß mit einem Buchstaben beginnen.
3	**Wenn der Funktionsbaustein Formaloperanden bearbeiten soll:** Geben Sie als Bausteinparameter die Formaloperanden ein, die Sie im Baustein verwenden. Für jeden Formaloperanden müssen Sie angeben: - den Namen des Bausteinparameters (maximal 4 Zeichen), - die Art des Bausteinparameters und evtl. den Typ des Bausteinparameters Sie können maximal 40 Formaloperanden definieren.
4	Geben Sie Ihr STEP-5-Programm in der Darstellungsart **AWL** ein. Die Formaloperanden werden dabei durch ein vorangestelltes Gleichheitszeichen gekennzeichnet (z. B. U =X1). Sie können auch mehrmals an verschiedenen Stellen im Funktionsbaustein angesprochen werden.
5	Schließen Sie die Programmeingabe mit "BE" (Bausteinende) ab.

Hinweis
Wenn Sie die **Reihenfolge** oder die **Anzahl** der Formaloperanden in der Formaloperandenliste ändern, müssen alle STEP-5-Anweisungen im Funktionsbaustein, die einen **Formaloperanden** ansprechen, und die Bausteinparameterliste im aufrufenden Baustein entsprechend nachgeführt werden!

Programmieren und ändern Sie Funktionsbausteine grundsätzlich auf Diskette oder Festplatte und übertragen Sie sie anschließend in die CPU!

Formaloperanden

Als Formaloperanden eines Funktionsbausteins (auch **Bausteinparameter** genannt) sind folgende Art- und Typbezeichnungen zugelassen:

Tabelle 2-5 Zulässige Formaloperanden für Funktionsbausteine

Parameterart	Parametertyp
E = Eingang A = Ausgang	BI/BY/W/D
D = Datum	KM/KH/KY/KC/KF/ KT/KZ/KG
B = Befehl T = Zeit (Timer) Z = Zähler	keine (Typangabe unzulässig)

E, D, B, T oder **Z** sind Parameterarten, die bei der graphischen Darstellung eines FB-Aufrufs auf der **linken** Seite des Funktionssymbols gezeichnet werden.
Mit **A** gekennzeichnete Parameter werden bei der grafischen Darstellung eines FB-Aufrufs auf der **rechten** Seite des Funktionssymbols gezeichnet.

Der Parametertyp gibt an, ob es sich bei E- und A-Parametern um Bit-, Byte-,Wort- oder Doppelwortgrößen handelt und welches Datenformat z. B. Bitmuster oder Hexadezimalmuster) für D-Parameter gilt.

2.3.3 Aufrufen und Parametrieren von Funktionsbausteinen

Jeder Funktionsbaustein kann beliebig oft und an beliebigen Stellen im STEP-5-Anwenderprogramm aufgerufen werden. Aufrufe von Funktionsbausteinen können sowohl in einer Anweisungsliste als auch in einer grafischen Darstellung erfolgen (FUP oder KOP).

Gehen Sie beim Aufrufen und Parametrieren folgendermaßen vor:

Schritt	Aktion	Reaktion am PG
1	Stellen Sie sicher, daß der aufgerufene Funktionsbaustein entweder im PG-Speicher (offline) oder im CPU-Speicher (online) vorhanden ist.	keine
2	Geben Sie im aufrufenden Baustein die Aufrufanweisung für den Funktionsbaustein ein. Der Aufruf eines Funktionsbausteins kann innerhalb eines Organisations-, Programm- oder Schrittbausteins oder innerhalb eines anderen Funktionsbausteins programmiert werden.	Nachdem Sie die Aufrufanweisung (z. B. SPA FB 200) eingegeben haben, erscheinen automatisch der Name und die Formaloperandenliste des betreffenden Funktionsbausteins.
3	Ordnen Sie jedem einzelnen Formaloperanden den für diesen Aufruf gültigen **Aktual**operanden zu . (Sie **parametrieren** dadurch den Funktionsbaustein.) Die Aktualoperanden können bei den einzelnen Aufrufen unterschiedlich sein: beim ersten Aufruf des FB 200 z. B. Ein- und Ausgänge, beim zweiten Aufruf Merker. Entsprechend der Formaloperandenliste müssen Sie bei jedem Aufruf eines Funktionsbausteins die erforderlichen Aktualoperanden zuordnen.	keine

Aufruf unbedingt/bedingt

unbedingter Aufruf	bedingter Aufruf
"SPA FBn" für Funktionsbausteine FB oder **"BA FXn"** für erweiterte Funktionsbausteine FX): Der angesprochene Funktionsbaustein wird unabhängig vom Verknüpfungsergebnis (VKE) bearbeitet.	**"SPB FBn"** für Funktionsbausteine FB oder **"BAB FXn"** für erweiterte Funktionsbausteine FX): Der angesprochene Funktionsbaustein wird nur dann bearbeitet, wenn das Verknüpfungsergebnis VKE = 1 ist. Bei VKE = 0 wird die Sprunganweisung nicht ausgeführt. Unabhängig davon, ob der Bausteinaufruf ausgeführt wird oder nicht, wird das VKE **immer auf '1' gesetz**t.
Nach dem unbedingten und bedingten Aufruf kann das Verknüpfungsergebnis nicht weiter verknüpft werden. Es bleibt beim Sprung in den FB erhalten und kann dort ausgewertet werden.	

Zulässige Aktualoperanden

Welche Operanden Sie einem Funktionsbaustein als **Aktualoperanden** zuordnen dürfen, entnehmen Sie bitte der nachfolgenden Tabelle.

Tabelle 2-6 Zulässige Aktualoperanden für Funktionsbausteine

Art des Parameters	**Typ des Parameters**		**Zugelassene Aktualoperanden**
E, A	BI	für einen Operanden mit Bitadresse	E n.m Eingang A n.m Ausgang M n.m Merker
	BY	für einen Operanden mit Byteadresse	EB n Eingangsbyte AB n Ausgangsbyte MB n Merkerbyte DL n Datenbyte links DRn Datenbyte rechts PY n Peripheriebyte QBn Byte der erweiterten Peripherie
	W	für einen Operanden mit Wortadresse	EW n Eingangswort AW n Ausgangswort MW n Merkerwort DW n Datenwort PW n Peripheriewort QW n Wort der erweiterten Peripherie
	D	für einen Operanden mit Doppelwortadresse	ED n Eingangs-Doppelwort AD n Ausgangs-Doppelwort MD n Merker-Doppelwort DD n Daten-Doppelwort
D	KM	für ein Binärmuster (16 Stellen)	Konstante
	KY	für zwei byteweise Betragszahlen im Bereich von jeweils 0 bis 255	
	KH	für ein Hexadezimalmuster bis 4 Stellen	
	KC	für zwei alphanumerische Zeichen	
	KT	für einen Zeitwert (BCD-codiert) mit Zeitraster .0 bis .3 und Zeitwert 0 bis 999	
	KZ	für einen Zählwert 0 bis 999	

Art des Parameters	Typ des Parameters	Zugelassene Aktualoperanden
Fortsetzung der Tabelle 2-6:		
D (Forts.)	KF für eine Festpunktzahl -32768 bis +32767 KG für eine Gleitpunktzahl [1)]	Konstante
B	keine Typangabe zulässig	DB n Datenbaustein; ausgeführt wird der Befehl A DB n FB n Funktionsbaustein (nur ohne Parameter zulässig) wird unbedingt (SPA . .n) aufgerufen OBn Organisationsbaustein wird unbedingt (SPA . .n) aufgerufen PB n Programmbaustein wird unbedingt (SPA . .n) aufgerufen SB n Schrittbaustein wird unbedingt (SPA . .n) aufgerufen
T	keine Typangabe zulässig	T 0 bis 255 Zeit
Z	keine Typangabe zulässig	Z 0 bis 255 Zähler

[1)] $\pm 0{,}1469368 \times 10^{-38}$ bis $\pm 0{,}1701412 \times 10^{39}$

Hinweis
S-Merker sind als Aktualoperanden für Funktionsbausteine **nicht** zugelassen.!

Nach dem Sprung in den Funktionsbaustein werden bei der Bearbeitung des Funktionsbaustein-Programms anstelle der Formaloperanden die Aktualoperanden aus dem aufrufenden Baustein verwendet.
Aufgrund dieser Eigenschaft der parametrierbaren Funktionsbausteine können Sie diese in Ihrem Anwenderprogramm vielseitig verwenden.

Beispiele

Beispiel 1: Folgendes (vollständiges) Beispiel soll Ihnen sowohl das Programmieren als auch das Aufrufen und Parametrieren eines Funktionsbausteins verdeutlichen. Sie können es selbst leicht nachvollziehen.

Der Funktionsbaustein FB 202 wird programmiert:

```
FB 202

NETZWERK 1
NAME BEISPIEL
BEZ   :  MONI    E/A/D/B/T/Z:  E     BI/BY/W/D:  BI
BEZ   :  BERT    E/A/D/B/T/Z:  E     BI/BY/W/D:  BI
BEZ   :  HANS    E/A/D/B/T/Z:  A     BI/BY/W/D:  BI

     :U=  MONI
     :U=  BERT
     :==  HANS

     :
     :  BE
```

Formal-operan-denliste

STEP-5-Anwei-sungen

Formal-operanden

Parameter-art

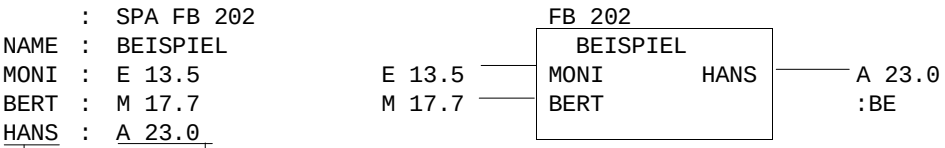
Parameter-typ

Der Funktionsbaustein FB 202 wird im Programmbaustein PB 25 aufgerufen und parametriert:

Darstellungsart AWL

```
PB 25
NETZWERK 1

     :  SPA FB 202
NAME :  BEISPIEL
MONI :  E 13.5
BERT :  M 17.7
HANS :  A 23.0
     :  BE
```

Darstellungsart KOP/FUP



Formal-operanden

Aktual-operanden

Folgende Operationen werden nach Sprung in den FB 202 ausgeführt:

```
: U   E 13.5
: U   M 17.7
: =   A 23.0
:BE
```

Beispiel 2: Aufruf und Parametrierung eines Funktionsbausteins mit den Darstellungsarten AWL und KOP/FUP in einem Programmbaustein

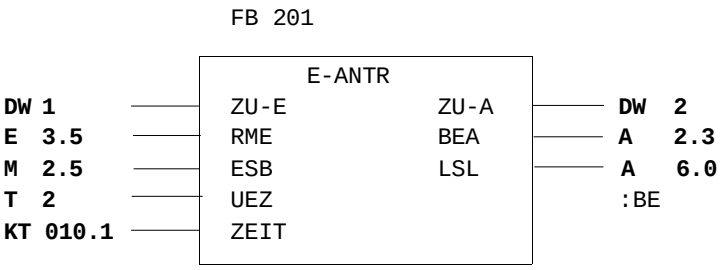
Darstellungsart AWL

```
PB 25
NETZWERK 1
      :
      :  A  DB 5
      :
      :  SPA   FB 201
NAME  :  E-ANTR
ZU-E  :        DW    1
RME   :        E     3.5
ESB   :        M     2.5
UEZ   :        T     2
ZEIT  :        KT    010.1
ZU-A  :        DW    2
BEA   :        A     2.3
LSL   :        A     6.0
      :  BE
```

Formal-operanden

Aktual-operanden

Darstellungsart KOP/FUP



2.3.4 Spezielle Funktionsbausteine

Neben den Funktionsbausteinen, die der Anwender selbst programmiert, gibt es die **Standard-Funktionsbausteine,** die als fertiges Softwareprodukt zu beziehen sind. Sie enthalten allgemein verwendbare Standardfunktionen (z. B. Meldefunktionen, Ablaufsteuerungen usw.). Standard-Funktionsbausteine belegen die Nummern FB 1 bis FB 199.

Wenn Sie Standard-Funktionsbausteine beziehen, beachten Sie die speziellen Hinweise in der dazugehörigen Beschreibung (belegte Bereiche, Konventionen usw.).

Die Standard-Funktionsbausteine für das AG S5-135U sind im Katalog ST 57 /11/ aufgeführt.

Beispiel

Gleitpunktradizierer RAD:GP FB 6

Der Funktionsbaustein RAD:GP radiziert eine Gleitpunktzahl (8-bit-Exponent und 24-bit-Mantisse), d. h. er bildet die Quadratwurzel. Das Ergebnis ist ebenfalls eine Gleitpunktzahl (8-bit-Exponent und 24-bit-Mantisse), wobei das niederwertigste Bit der Mantisse nicht gerundet wird.

Der Funktionsbaustein setzt für die weitere Verarbeitung gegebenenfalls die Kennung "Radikand negativ".

Zahlenbereich:

Radikand	- 0,1469368 Exp. -38	bis +0,1701412 Exp. +39
Wurzel	+0,3833434 Exp. -19	bis +0,1304384 Exp. +20

Funktion: $Y = \sqrt{A}$

Y = SQRT; A = RADI

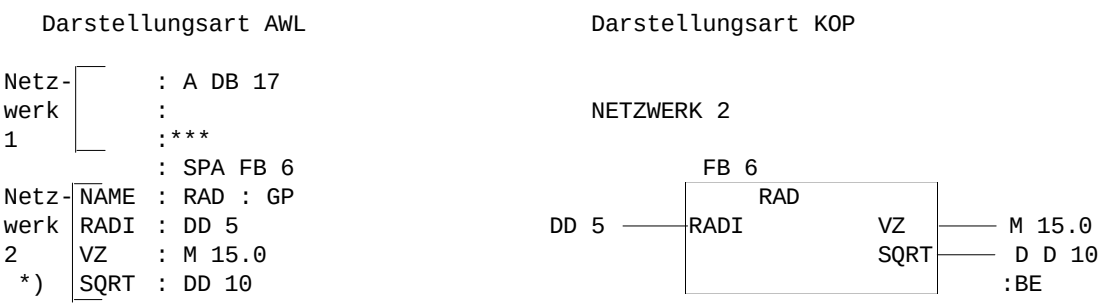
Aufruf des Funktionsbausteins FB 6:

Im aufgeführten Beispiel wird eine Gleitpunktzahl, die im DD 5 des DB 17 mit 8-bit-Exponent und 24-bit-Mantisse bereitgestellt ist, radiziert. Das Ergebnis, wieder eine 32-bit-Gleitpunktzahl, wird im DD 10 abgelegt. Vorher muß der entsprechende Datenbaustein aufgeschlagen werden. Der Parameter VZ (Parameterart: A, Parametertyp: BI) gibt das Vorzeichen des Radikanden an: VZ = 1 bei negativem Radikanden.

Belegte Merkerwörter: MW 238 bis 254.

Fortsetzung auf der nächsten Seite

Fortsetzung "Gleitpunktradizierer":



DD= Datendoppelwort

*) Muß in getrennten Netzwerken stehen, da der Befehl "A DB 17" in Netzwerk 1 nicht umsetzbar in KOP/FUP ist.

Benutzung des FB 0

Wenn der Organisationsbaustein OB 1 nicht programmiert ist, ruft das Systemprogramm anstelle des OB 1 zyklisch den FB 0 auf, sofern dieser geladen ist.

Da Sie in einem Funktionsbaustein den gesamten Operationsvorrat der Programmiersprache STEP 5 zur Verfügung haben, eignet sich die Programmierung des FB 0 – anstelle des OB 1 – besonders dann, wenn Sie ein kurzes und zeitkritisches Programm bearbeiten lassen wollen.

Hinweis
Der FB 0 sollte deshalb nur zur Programmierung der **zyklischen** Programmbearbeitung verwendet werden! (Er darf keine Parameter enthalten.)
Sind sowohl OB 1 als auch FB 0 geladen, so wird nur der Organisationsbaustein **OB 1 zyklisch** vom Systemprogramm aufgerufen.

2.4 Datenbausteine

In Datenbausteinen (DB) oder erweiterten Datenbausteinen (DX) sind die festen oder variablen Daten abgelegt, mit denen das Anwenderprogramm arbeitet. In Datenbausteinen werden **keine** STEP-5-Operationen bearbeitet.

Die Daten eines Datenbausteins können sein:

- beliebige Bitmuster, z. B. für Anlagenzustände,
- Zahlen (hexadezimal, dual, dezimal) für Zeitwerte, Rechenergebnisse,
- alphanumerische Zeichen, z. B. für Meldetexte.

Aufbau eines Datenbausteins

Ein Datenbaustein (DB/DX) besteht aus den Teilen

- Bausteinvorkopf (DV, DXV),
- Bausteinkopf und
- Bausteinrumpf.

Bausteinvorkopf

Der **Bausteinvorkopf** wird automatisch auf der Festplatte oder der Diskette des PG angelegt und nicht in die CPU übertragen. Er enthält die Datenformate der im Bausteinrumpf eingegebenen Datenwörter. Als Anwender haben Sie keinen Einfluß auf das Anlegen des Bausteinvorkopfes.

Hinweis
Wenn Sie einen Datenbaustein aus der CPU auf Diskette oder auf Festplatte übertragen, wird der dazugehörige Bausteinvorkopf gelöscht. Aus diesem Grund dürfen Sie einen Datenbaustein mit unterschiedlichen Datenformaten nie in der CPU ändern und ihn anschließend auf die Diskette zurückübertragen, sonst wird allen Datenwörtern dieses DB automatisch das Datenformat zugeordnet, das Sie in der Voreinstellungsmaske gewählt haben.

Bausteinkopf

Der **Bausteinkopf** belegt 5 Wörter im Speicher und enthält

- die Bausteinkennung,
- die Kennung des Programmiergerätes,
- den Bausteintyp und die Bausteinnummer,
- die Bibliotheksnummer und
- die Bausteinlänge (inkl. Länge des Bausteinkopfes).

Bausteinrumpf

Der **Bausteinrumpf** enthält in aufsteigender Reihenfolge, beginnend mit Datenwort DW 0, die Datenwörter, mit denen das Anwenderprogramm arbeitet. Jedes Datenwort belegt im Speicher 1 Wort (16 bit).

Maximale Länge

Datenbausteine dürfen insgesamt bis zu 4096 Wörter (inkl. Kopf) im Speicher der CPU belegen. Berücksichtigen Sie beim Eingeben und Übertragen von Datenbausteinen mit dem PG den Speicherausbau Ihrer CPU!

2.4.1 Erstellen von Datenbausteinen

So erstellen Sie einen Datenbaustein:

Schritt	Aktion
1	Geben Sie den Bausteintyp (DB/DX) und eine Datenbaustein-**Nummer** zwischen 3 und 255 ein.
2	Geben Sie die einzelnen **Datenwörter** im gewünschten Datenformat ein. (Die Eingabe der Datenwörter wird **nicht** mit einer BE-Anweisung abgeschlossen!)

Hinweis
Die Datenbausteine DB 0, DB 1, DB 2, DX 0 DX 1 und DX 2 sind für bestimmte Funktionen reserviert und damit nicht frei verwendbar (siehe "Spezielle Datenbausteine")!

Tabelle 2-7 In einem Datenbaustein zulässige Datenformate

Bezeichnung	Datenformat	Beispiel
KM	Bitmuster	00100110 00111111
KH	Hexadezimalzahl	263F
KY	Byte	038,063
KF	Festpunktzahl	09791
KG	Gleitpunktzahl	+1356123+12
KC	Zeichen	?!ABCD123-+.,%
KT	Zeitwert eines Zeitglieds	055.2
KZ	Zählerwert	234

2.4.2 Aufschlagen von Datenbausteinen

Ein Datenbaustein (DB/DX) kann nur unbedingt aufgeschlagen werden. Dies ist möglich innerhalb eines Organisations-, Programm-, Schritt- oder Funktionsbausteins. Ein bestimmter Datenbaustein kann mehrfach im Programm aufgeschlagen werden.

So schlagen Sie Datenbausteine auf:

WENN ...	DANN ...
Sie einen **DB**-Datenbaustein aufschlagen wollen	geben Sie die STEP-5-Operation **"A DB.."** ein
Sie einen **DX**-Datenbaustein aufschlagen wollen	geben Sie die STEP-5-Operation **"AX DX.."** ein

Gültigkeitsbereich

Nach Aufschlagen eines Datenbausteins beziehen sich alle folgenden Anweisungen mit dem Operandenbereich **'D'** auf den aufgeschlagenen Baustein.

Der aufgeschlagene Datenbaustein bleibt auch dann gültig, wenn durch einen Bausteinaufruf die Programmbearbeitung in einem anderen Baustein fortgesetzt wird.

Wenn in diesem Baustein nun ein anderer Datenbaustein aufgeschlagen wird, ist dieser **nur** im aufgerufenen Baustein gültig. Nach Rücksprung in den aufrufenden Baustein gilt wieder der alte Datenbaustein.

Zugriff

Der **Zugriff** auf die in dem aufgeschlagenen Datenbaustein gespeicherten Daten erfolgt bei der Programmbearbeitung durch die **Lade**- und **Transferoperationen** (Einzelheiten dazu lesen Sie bitte in Kapitel 3).

Mit einer **binären Verküpfung** wird das adressierte Datenwort-Bit zur Bildung des VKE herangezogen. Der Inhalt des Datenwortes wird nicht verändert.

Bei einer **Speicheroperation** wird dem adressierten Datenwort-Bit der Wert des VKE zugewiesen. Der Inhalt des Datenwortes kann dabei verändert werden.

Mit einer **Ladeoperation** wird der Inhalt des adressierten Datenwortes in den AKKU 1 übertragen. Der Inhalt eines Datenwortes nicht verändert.

Mit einer **Transferoperation** werden Daten aus dem AKKU 1 in das adressierte Datenwort übertragen. Der alte Inhalt eines Datenwortes wird überschrieben.

Hinweis
Vor dem Zugriff auf ein Datenwort müssen Sie im Anwenderprogramm den gewünschten Datenbaustein aufschlagen, da die CPU nur so das richtige Datenwort findet.
Das adressierte Datenwort muß im aufgeschlagenen Baustein enthalten sein, sonst erkennt das Systemprogramm bei einem Zugriff einen Lade- bzw. Transferfehler.

Mit Lade- und Transferoperationen können Sie nur bis zu Datenwortnummer 255 zugreifen!

Ein aufgeschlagener Datenbaustein bleibt gültig bis

a) ein anderer Datenbaustein aufgeschlagen wird

oder

b) ein aufgerufener Baustein mit ’BE’ oder ’BEB’ beendet wird.

Beispiele

Beispiel 1: Transferieren von Datenwörtern

Es soll der Inhalt des Datenwortes DW 1 vom Datenbaustein DB 10 in das Datenwort DW 1 des Datenbausteins DB 20 transferiert werden.

Dazu geben Sie folgende Anweisungen ein:

```
   :A    DB 10    (DB 10 aufschlagen)
   :L    DW 1     (Inhalt DW 1 in den AKKU 1
   :              übertragen)
   :A    DB 20    (DB 20 aufschlagen)
   :T    DW 1     (Inhalt AKKU 1 nach  DW 1
   :              übertragen)
   :
```

Beispiel 2: Gültigkeitsbereich bei Datenbausteinen (Bild 2-5)

Im Programmbaustein PB 7 wird der Datenbaustein DB 10 aufgeschlagen (A DB 10). In der folgenden Programmbearbeitung werden die Daten dieses Datenbausteins bearbeitet.

Nach dem Aufruf (SPA PB 20) wird der Programmbaustein PB 20 bearbeitet. Der Datenbaustein DB 10 ist jedoch nach wie vor gültig. Erst mit dem Aufschlagen des Datenbausteins DB 11 (A DB 11) wird der Datenbereich gewechselt. Bis zum Ende des Programmbausteins PB 20 (BE) ist nun der Datenbaustein DB 11 gültig.

Nach Sprung zurück in den Programmbaustein PB 7 ist wieder der Datenbaustein DB 10 gültig.

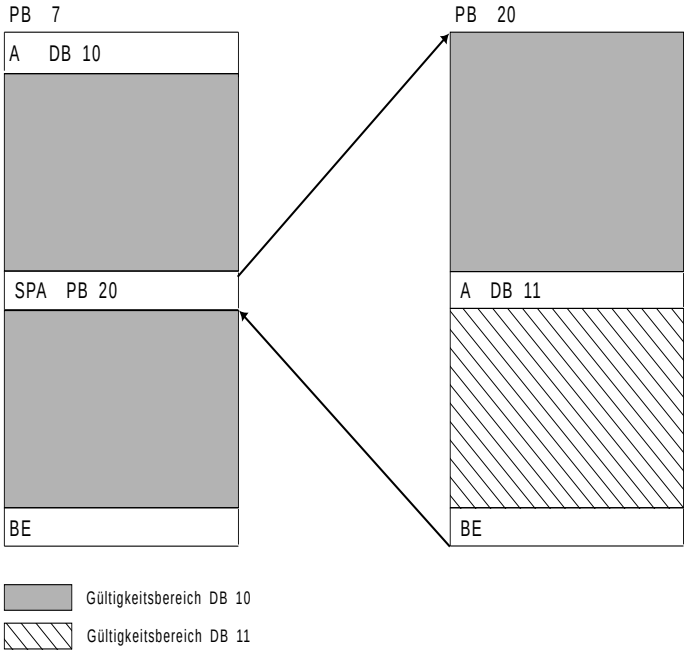


Bild 2-5 Gültigkeitsbereich eines aufgeschlagenen Datenbausteins

2.4.3 Spezielle Datenbausteine

Die Datenbausteine DB 0, DB 1, DB 2, DX 0, DX 1 und DX 2 sind bei der CPU 928B für bestimmte Funktionen reserviert. Sie werden vom Systemprogramm verwaltet und sind für den Anwender nicht beliebig verwendbar.

DB 0

- **Datenbaustein DB 0** (siehe Abschnitt 8.3.2)

  Der Datenbaustein DB 0 enthält die Adreßliste mit den Anfangsadressen aller Bausteine, die sich im Anwenderspeicher oder im Datenbaustein-RAM der CPU befinden. Diese Adreßliste wird vom Systemprogramm bei der Initialisierung (nach jedem NETZ EIN und nach URLÖSCHEN) erzeugt und bei der Eingabe oder Änderung von Bausteinen mit dem PG automatisch aktualisiert.

DB 1

- **Datenbaustein DB 1** (siehe Abschnitt 10.1.6)

  Der Datenbaustein DB 1 enthält die Liste der digitalen Ein- und Ausgänge (P-Peripherie mit relativen Byteadressen von 0 bis 127) sowie der Koppelmerkerein- und -ausgänge, die der CPU zugeordnet sind, und gegebenenfalls eine Zeitenblocklänge.

  DB 1 **kann** parametriert und geladen werden:

  Um die Zykluszeit im Einzelprozessorbetrieb zu verringern, da nur die im DB 1 eingetragenen Ein- und Ausgänge oder Zeiten aktualisiert werden.

  DB 1 **muß** parametriert und geladen werden:

  a) bei Mehrprozessorbetrieb.
  b) wenn Koppelmerker mit CPs vorhanden sind.

DB 2

- Datenbaustein DB 2 (siehe Abschnitt 4.4.3)

  Der Datenbaustein DB 2 dient zur Parametrierung der Reglerstruktur R64 durch den Anwender. Die Regelungsfunktion können Sie als Softwareprodukt beziehen. Sie wird vom Systemprogramm unterstützt.

DX 0

- **Datenbaustein DX 0** (siehe Kapitel 7)

  Durch Parametrieren und Laden des Datenbausteins DX 0 können Sie die Voreinstellungen bestimmter Systemprogrammfunktionen (z. B. bei der Bearbeitung des Anlaufs) ändern und damit die Leistungen des Systemprogramms Ihren Erfordernissen anpassen.

DX 1

- **Datenbaustein DX 1**

  Reserviert.

DX 2

- Der Datenbaustein DX 2 wird dazu verwendet, um für die Kommunikation über die 2. serielle Schnittstelle den Kopplungstyp festzulegen. Einzelheiten zur Parametrierung dieses Bausteins finden Sie im Handbuch "Kommunikation" /14/.

Programmbearbeitung 3

Inhalt von Kapitel 3

Programmbearbeitung

3

Dieses Kapitel wendet sich an Leser, die in der Anwendung der Programmiersprache noch keine große Erfahrung haben. Es führt daher in die Gundlagen der STEP-5-Programmierung ein und erläutert im weiteren ausführlich (mit Beispielen) die STEP-5-Operationen der CPU 928B.

Erfahrenen Lesern, denen die Information zu einer speziellen STEP-5-Operation im Tabellenheft /1/ nicht ausreicht, kann der Abschnitt 3.5 als Nachschlageteil dienen.

3.1 Prinzip der Programmbearbeitung

Das STEP-5-Anwenderprogramm kann auf verschiedene Art und Weise bearbeitet werden.

Typischerweise herrscht bei speicherprogrammierbaren Steuerungen (SPS) die zyklische Programmbearbeitung vor: Das Systemprogramm läuft in einer Programmschleife (dem Zyklus, siehe Abschnitt 3.4) und ruft dabei in jeder Schleife einmal den Organisationsbaustein OB 1 auf (siehe Bild 3-1).

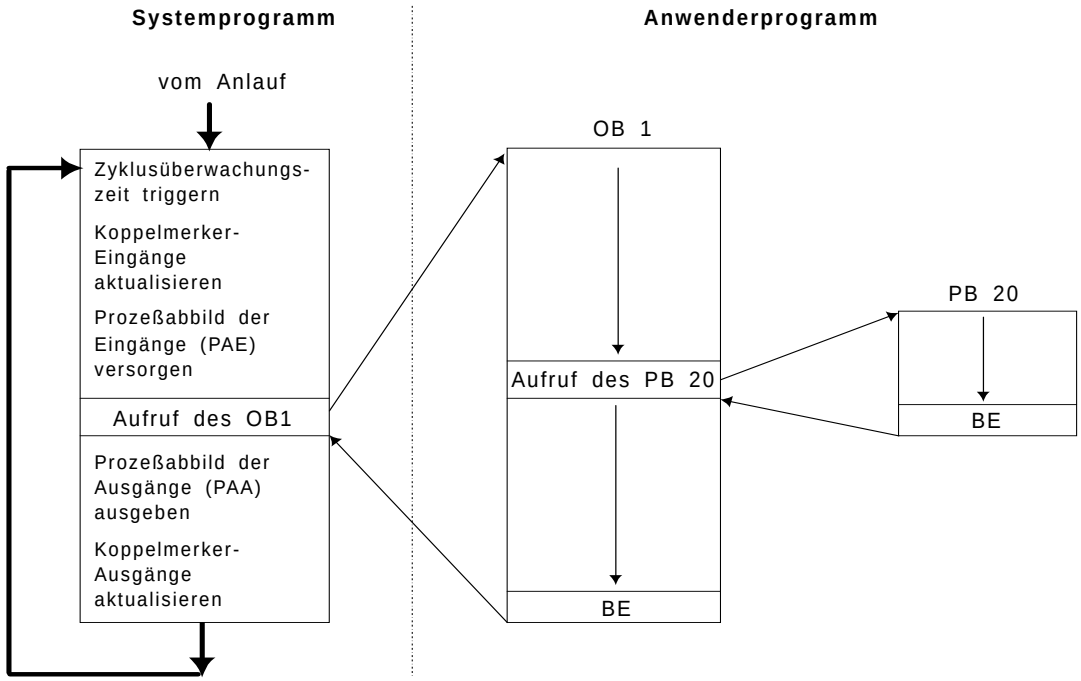


Bild 3-1 Prinzip der zyklischen Programmbearbeitung

3.2 Programmorganisation

Mit der Programmorganisation legen Sie fest, nach welchen Bedingungen und in welcher Reihenfolge die von Ihnen erstellten Bausteine bearbeitet werden sollen. Dazu programmieren Sie in den Organisationsbausteinen bedingte oder unbedingte Aufrufe der gewünschten Bausteine.

In den Programmteilen der einzelnen Organisations-, Programm-, Funktions- und Schrittbausteine können weitere Programm-, Funktions- und Schrittbausteine in beliebiger Kombination (nacheinander oder ineinander verschachtelt) aufgerufen werden.

Das Anwenderprogramm sollte zweckmäßigerweise so organisiert sein, daß es die wesentlichen Programmstrukturen oder programmtechnisch zusammenhängende Anlagenteile hervorhebt.

Die Bilder 3-2 und 3-3 zeigen Ihnen zwei Beispiele einer Programmstruktur.

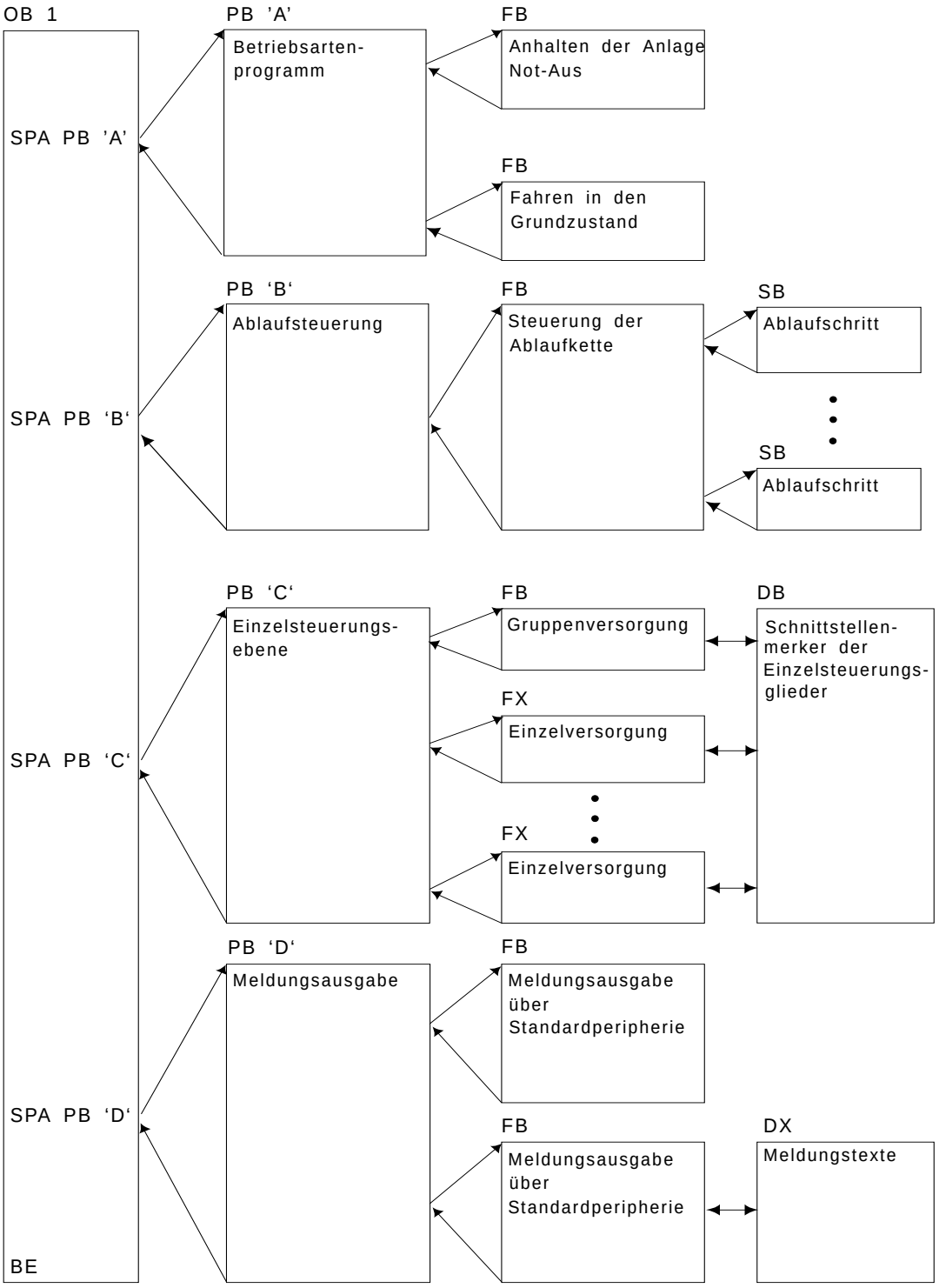


Bild 3-2 Beispiel für die Organisation des Anwenderprogramms nach Programmstruktur

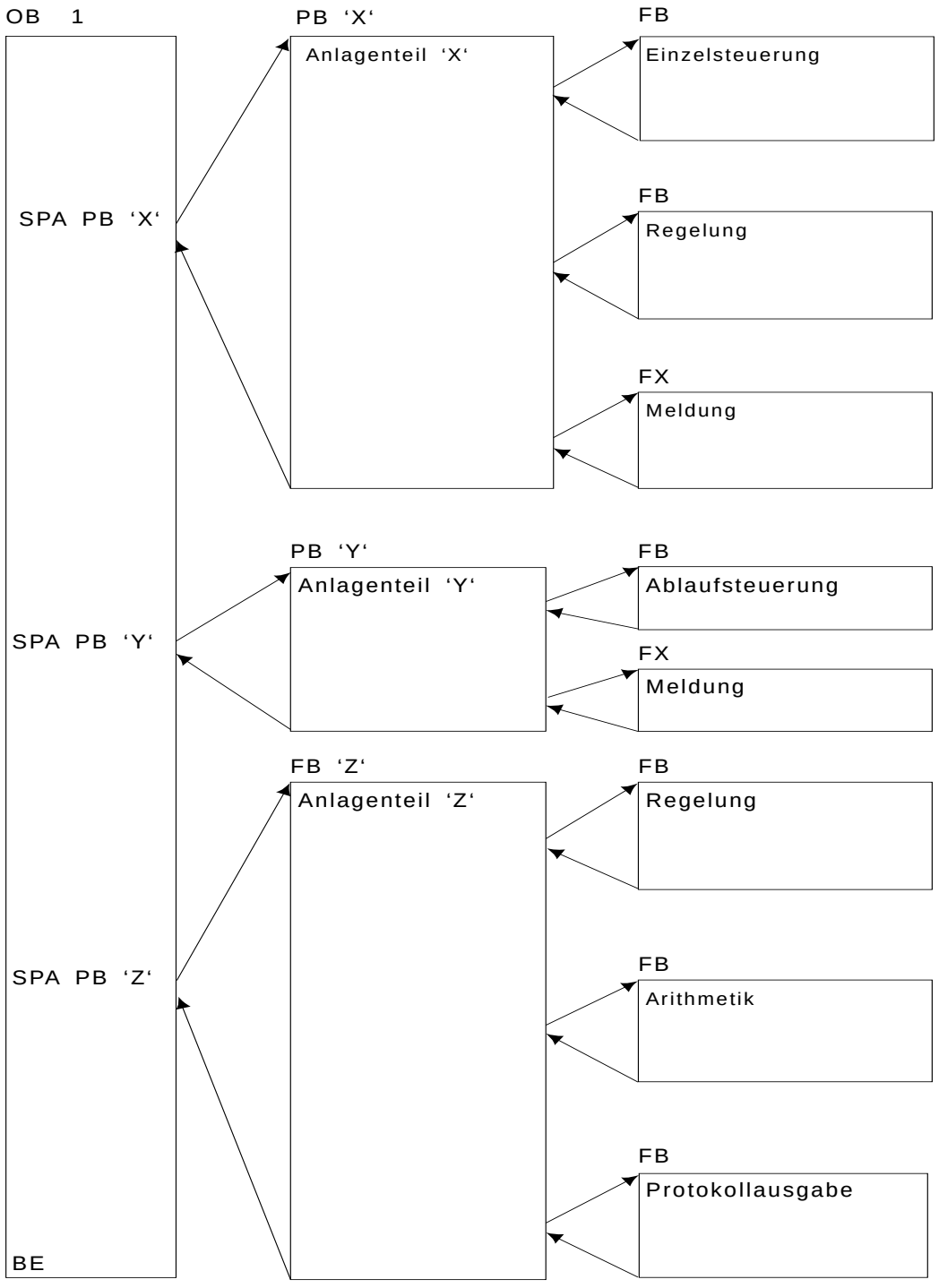
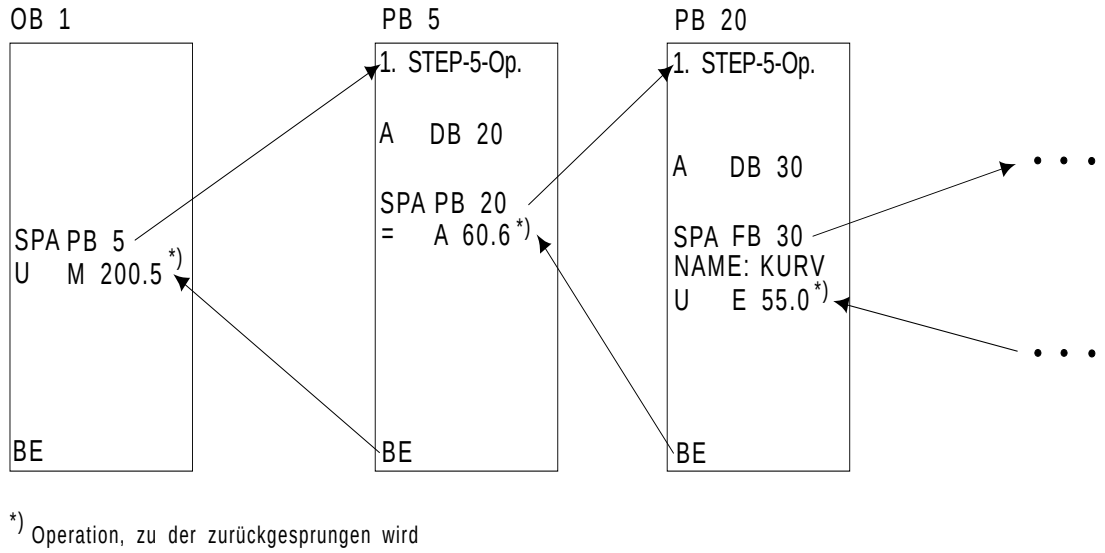


Bild 3-3 Beispiel für die Organisation des Anwenderprogramms nach Anlagenstruktur

Bausteinschachtelung Bild 3-4 zeigt Ihnen das Prinzip der verschachtelten Bausteinaufrufe.



*) Operation, zu der zurückgesprungen wird

Bild 3-4 Verschachteltes Aufrufen von Codebausteinen

Bausteinadressen

Die Lage eines Bausteins im Anwenderspeicher (oder DB-RAM) ist festgelegt durch seine Baustein-Anfangsadresse: Dies ist bei Code-Bausteinen die Adresse derjenigen Zelle im Speicher, in der sich die erste STEP-5-Operation des Bausteins befindet (bei FB und FX der SPA-Befehl über die Formaloperandenliste); bei Datenbausteinen ist es die Adresse des ersten Datenwortes.

Damit die CPU bei einem Bausteinaufruf den aufgerufenen Baustein im Speicher findet, sind die Anfangsadressen aller gültigen Bausteine in der Bausteinadreßliste im Datenbaustein DB 0 eingetragen. Der DB 0 wird vom Systemprogramm verwaltet, als Anwender können Sie ihn nicht aufschlagen!

Um nach Abarbeitung des aufgerufenen Bausteins den Rückweg in den aufrufenden Baustein zu finden, speichert die CPU bei jedem Aufruf eines neuen Bausteins die **Rücksprungadresse**: Die Rücksprungadresse ist die Adresse derjenigen Zelle im Speicher, in der die dem Bausteinaufruf folgende STEP-5-Anweisung steht. Außerdem wird von der CPU die **Anfangsadresse und Länge des Datenbausteins** gespeichert, der an dieser Stelle gültig ist.

Schachteltiefe

Sie können maximal 62 Bausteine ineinander schachteln. Werden mehr als 62 Bausteine aufgerufen, meldet die CPU einen Fehler und geht in den Stoppzustand.

Beispiel

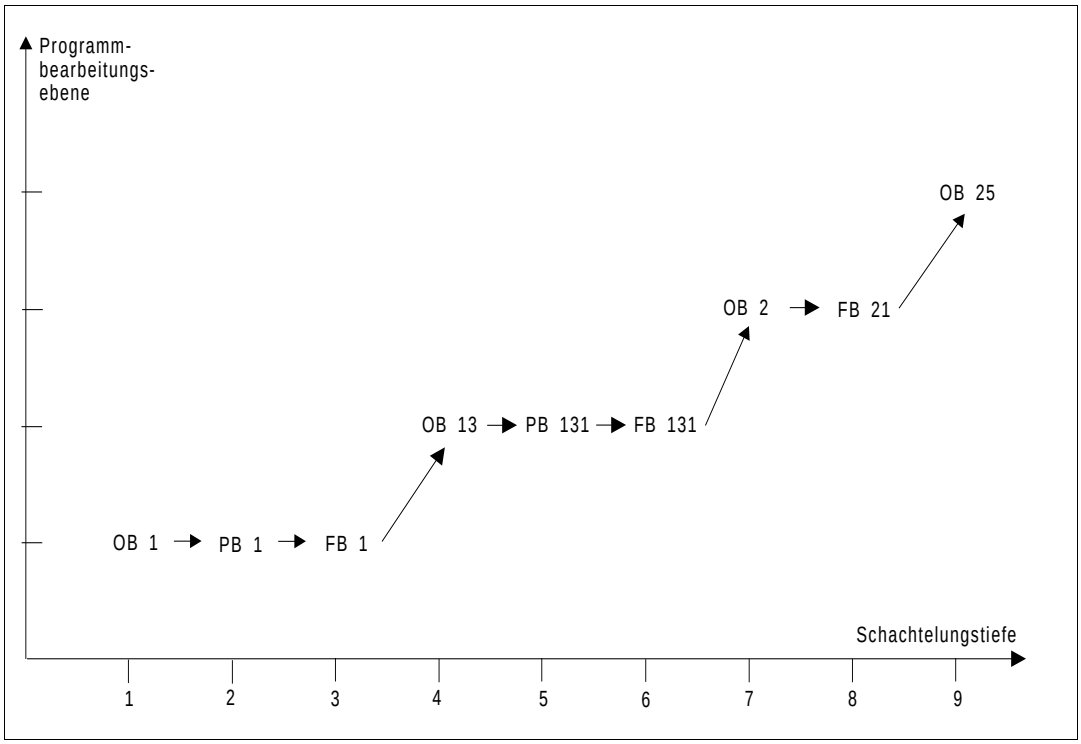


Bild 3-5 Beispiel für Bausteinschachtelungstiefe

So ermitteln Sie die Schachtelungstiefe Ihres Programms :

- Addieren Sie alle von Ihnen programmierten Organisationsbausteine (im Beispiel: 4 OB).

- Addieren Sie die Schachtelungstiefen der einzelnen Organisationsbausteine (im Beispiel: 2 + 2 + 1 + 0 = 5).

- Beide Beträge zusammen ergeben die Programm-Schachtelungstiefe (im Beispiel: 4 + 5 = Schachelungstiefe 9). Sie darf den Wert 62 nicht überschreiten

3.3 Speicherung von Programm- und Datenbausteinen

Damit die CPU das Anwenderprogramm bearbeiten kann, muß dieses – einschließlich vorhandener Datenbausteine – in den Programmspeicher geladen werden. Als Programmspeicher stehen Ihnen das Anwendermodul (wahlweise RAM oder EPROM) und das DB-RAM zur Verfügung.

Verschiedene Arten der Speicherung [1)]

- **RAM-Modul:** Wenn Sie ein steckbares RAM-Modul verwenden, können Sie Ihr Anwenderprogramm direkt vom Programmiergerät in die CPU übertragen.
  Bei einem RAM-Modul läßt sich der Speicherinhalt schnell ändern. Die Pufferbatterie des Zentralgeräts verhindert, daß - wenn das RAM-Modul gesteckt ist - bei NETZ AUS das Anwenderprogramm im Speicher gelöscht wird.
  Alle programmierten Bausteine werden in beliebiger Reihenfolge im RAM-Modul gespeichert (siehe Kapitel 2.1.5, Bild 2.2). Sobald Sie einen Baustein ändern, ändert sich auch die Reihenfolge der Bausteine im Speicher.

  Wenn Sie zum Speichern Ihres Anwenderprogramms ein **RAM-Modul mit Pufferbatterie** verwenden, können Sie es aus der CPU entfernen, ohne daß Daten verloren gehen. Eine eigene Batterie schützt das Modul vor Datenverlust und stellt sicher, daß die Daten bis zum nächsten Einsatz erhalten bleiben.

- **EPROM-Modul:** Das gesamte Anwenderprogramm wird in einem steckbaren EPROM- Modul fest hinterlegt. In einem EPROM-Modul ist das Anwenderprogramm auch bei NETZ AUS und fehlender Pufferbatterie voll geschützt.
  Der Inhalt eines EPROM-Moduls kann über die CPU nicht verändert werden. Aus diesem Grund müssen Datenbausteine, die variable Daten enthalten und im Ablauf des Anwenderprogramms geändert werden sollen, im 1. NEUSTART nach URLÖSCHEN aus dem EPROM-Modul in das Datenbaustein-RAM der CPU kopiert werden. Dies müssen Sie entsprechend programmieren (siehe Sonderfunktions-OB 254 und 255, Kapitel 6.4.6).

- **DB-RAM:** In das DB-RAM werden Datenbausteine DB/DX durch Erzeugen oder Kopieren geschrieben. Beim Übertragen von Datenbausteinen vom PG in die CPU werden diese im DB-RAM abgelegt, falls das RAM-Modul gefüllt oder ein EPROM-Modul gesteckt ist.

Vorsicht
Gepufferte RAM-Module dürfen nicht über die EPROM-Schnittstelle des PG programmiert werden, da sie dabei zerstört werden können!

[1)] Beachten Sie bei der Speicherung von Datenbausteinen auch das "alternative Laden" – Abschnitt 2.1.5.

3.4 Bearbeitung des Anwenderprogramms

Die gesamte Software auf der CPU (diese besteht aus dem Systemprogramm und dem STEP-5-Anwenderprogramm) hat folgende Aufgaben zu bearbeiten:

- ANLAUF der CPU,
- Steuerung eines Automatisierungsprozesses durch ständig sich wiederholende Operationsfolgen (ZYKLUS),
- Steuerung eines Automatisierungsprozesses durch Reaktion auf Ereignisse, die sporadisch oder zu bestimmten Zeiten eintreffen (Alarme) sowie Reaktion auf Fehler.

Bei allen drei Aufgaben haben Sie die Möglichkeit, über Anwenderschnittstellen (Organisationsbausteine OB 1 bis OB 35 – siehe Abschnitt 2.2.3) spezielle Teile Ihres Anwenderprogramms auf der CPU ablaufen zu lassen.

ANLAUF

Bevor die CPU in die zyklische Bearbeitung eintreten kann, ist es u. U. erforderlich, eine Initialisierung durchzuführen, um einen definierten Ausgangszustand für die zyklische Bearbeitung herzustellen und z. B. Zeitraster für die Ausführung bestimmter Funktionen vorzugeben. Welche Initialisierung dies ist, hängt von dem Ereignis ab, das zu einem ANLAUF führt, sowie von Einstellungen, die Sie an Ihrer CPU vornehmen. Nähere Erläuterungen hierzu finden Sie in Kapitel 4.

Sie können das Verhalten der CPU bei einem ANLAUF beeinflussen durch Programmieren der Organisationsbausteine OB 20, OB 21 und OB 22 oder durch Parametrieren des Datenbausteins DX 0 (siehe Kapitel 7).

ZYKLUS

Nachdem ein ANLAUF durchgeführt worden ist, tritt das Systemprogramm in die zyklische Bearbeitung ein. Es übernimmt dabei Hintergrundfunktionen, die für Automatisierungsaufgaben erforderlich sind (siehe Bild 3-1 am Kapitelanfang).
Nach Ausführung der Systemfunktionen zu Beginn eines ZYKLUS wird vom Systemprogramm als zyklisches Anwenderprogramm der Organisationsbaustein OB 1 oder der Funktionsbaustein FB 0 aufgerufen. In einem dieser Bausteine programmieren Sie die STEP-5-Operationen für die zyklische Bearbeitung.

Reaktion bei Alarmen und Fehlern

Um auf einen Alarm oder Fehler speziell reagieren zu können, stehen Ihnen auf der CPU 928B spezielle Organisationsbausteine (OB 2, OB 6 und OB 9 bis OB 18 für Alarmbearbeitung, OB 19 und OB 23 bis OB 35 für Reaktionen im Fehlerfall) zur Verfügung, in denen Sie das entsprechende STEP-5-Programm hinterlegen können.

Bei der Alarm- oder Fehlerbearbeitung wird vom Systemprogramm der zugehörige Organisationsbaustein in die zyklische Bearbeitung "eingeschachtelt". Dies bedeutet, daß die zyklische Bearbeitung durch die Bearbeitung eines Alarms oder Fehlers unterbrochen wird. Die Einschachtelung der entsprechenden Organisationsbausteine erfolgt nach einem festen Prioritätenschema (weitere Informationen hierzu finden Sie in den Kapiteln 4 und 5).

Außer durch die genannten Organisationsbausteine können Sie das Verhalten der CPU bei der Alarmbearbeitung und im Fehlerfall durch Parametrieren des Datenbausteins DX 0 beeinflussen.

Die Organisationsbausteine OB 1 bis OB 39 können vom Systemprogramm aufgerufen werden, sobald sie in den Programmspeicher geladen worden sind **(auch im laufenden Betrieb)**.
Werden sie nicht geladen, so erfolgt entweder keine Reaktion der CPU oder – in den meisten Fehlerfällen – sie geht in den Stoppzustand (siehe hierzu Abschnitt 5.4).

Wie die Organisationsbausteine läßt sich auch der Datenbaustein DX 0 im laufenden Betrieb in den Programmspeicher laden. **Er wird jedoch erst mit dem nächsten NEUSTART wirksam.** Wird der DX 0 nicht geladen, gelten die Standardeinstellungen (siehe Kapitel 7).

3.4.1 Begriffsdefinitionen für die Programmbearbeitung

Zykluszeit

Der Zyklus beginnt mit der Triggerung der sogenannten Zykluszeitüberwachung und endet bei der nächsten Triggerung. Die Zeit, die die CPU für die Programmbearbeitung zwischen zwei Triggerungen benötigt, wird Zykluszeit genannt. Sie setzt sich aus der Laufzeit des Systemprogramms und der Laufzeit des Anwenderprogramms zusammen.

In die Zykluszeit geht somit ein:

- die Bearbeitungszeit des zyklischen Programms (System- und Anwenderprogramm),
- die Bearbeitungszeit von Alarmen (z. B. zeitgesteuerter Alarm),
- die Bearbeitungszeit von Unterbrechungen (Fehler).

Zykluszeitüberwachung

Die Zykluszeit wird von der CPU auf einen Maximalwert überwacht. Standardmäßig ist dieser Maximalwert auf 150 ms eingestellt. Als Anwender haben Sie die Möglichkeit, die Zykluszeitüberwachung selbst einzustellen bzw. sie während der Anwenderprogrammbearbeitung neu zu starten (siehe DX 0/Kapitel 7 und Sonderfunktions-OB OB 221 und OB 222/Abschnitte 6.22 und 6.23).

Prozeßabbild der Ein- und Ausgänge (PAE und PAA)

Vor Beginn der zyklischen Anwenderprogrammbearbeitung liest das Systemprogramm die Signalzustände der Eingabe-Peripheriebaugruppen in das Prozeßabbild der Eingänge ein. Das Anwenderprogramm wertet die Signalzustände im Prozeßabbild der Eingänge aus und setzt in Abhängigkeit dieser Auswertung die Signalzustände für die Ausgänge im Prozeßabbild der Ausgänge. Nach Bearbeitung des Anwenderprogramms überträgt das Systemprogramm die Signalzustände des Prozeßabbilds der Ausgänge zu den Ausgabe-Peripheriebaugruppen.

Durch die Zwischenspeicherung der Peripheriesignale im Prozeßabbild der Ein- und Ausgänge wird vermieden, daß ein Ändern des logischen Zustandes eines Bits innerhalb eines Programmzyklus zum "Flattern" des zugehörigen Peripherieausgangs führt.

Das Prozeßabbild ist somit ein Speicherbereich, dessen Inhalt nur **einmal pro Zyklus** an die Peripherie ausgegeben bzw. von der Peripherie eingelesen wird.

Hinweis
Ein Prozeßabbild existiert nur für Ein- und Ausgabebytes der P-Peripherie mit Byteadressen von 0 bis 127!

Koppelmerker

Koppelmerker dienen zum Datenaustausch zwischen den einzelnen CPUs im Mehrprozessorbetrieb bzw. zwischen der CPU und einigen Kommunikationsprozessoren.

Das Systemprogramm liest vor Beginn der zyklischen Anwenderprogrammbearbeitung die Eingangskoppelmerker der CPU ein und überträgt nach Bearbeitung des Anwenderprogramms die Ausgangskoppelmerker zum Koordinator bzw. zu den Kommunikationsprozessoren.

Die Definition der Ein- und Ausgangskoppelmerker erfolgt durch Erstellen des Datenbausteins DB 1 (siehe Abschnitt 10.1.6).

Unterbrechungsereignisse

Die zyklische Programmbearbeitung kann unterbrochen werden durch

- eine prozeßalarmgesteuerte Programmbearbeitung,
- eine zeitgesteuerte Programmbearbeitung,
- einen Verzögerungsalarm,
- einen uhrzeitgesteuerten Weckalarm.

Sie kann unterbrochen bzw. ganz abgebrochen werden

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch eine Stopp-Operation.

3.5 STEP-5-Operationen mit Beispielen

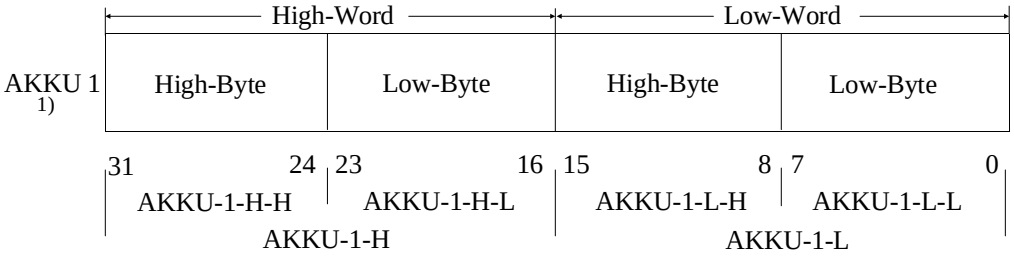
Eine STEP-5-Operation setzt sich zusammen aus einem Operationsteil und einem Operanden. Im Operationsteil wird festgelegt, **was** durch die CPU getan werden soll (Operation). Der Operandenteil gibt an, **womit** eine Operation durchgeführt werden soll.

Die STEP-5-Operationen lassen sich in verschiedene Gruppen einteilen:

- **Grundperationen** (in **allen** Codebausteinen anwendbar),
- **Ergänzende Operationen** (nur in Funktionsbausteinen FB/FX anwendbar),
- **Organisatorische Operationen** (nur in Funktionsbausteinen FB/FX anwendbar),
- **Semaphor-Operationen** (nur in Funktionsbausteinen FB/FX anwendbar).

Akkumulatoren als Arbeitsregister

Die CPU 928B besitzt vier Akkumulatoren, AKKU 1 bis AKKU 4. Ein überwiegender Teil der STEP-5-Operationen verwendet als Quelle für die Operanden und als Ziel für die Ergebnisse zwei Register (32 bit): AKKU 1 und AKKU 2.



Die Akkumulatoren werden abhängig von der auszuführenden STEP-5-Operation beeinflußt, z. B.:

- bei den Ladeoperationen wird als Ziel immer der AKKU 1 verwendet; der alte Inhalt von AKKU 1 wird in den AKKU 2 (Stack Lift) geschoben; die Akkumulatoren 3 und 4 werden bei allen Ladeoperationen nicht verändert,

1) bei AKKU 2 bis AKKU 4 analog

- arithmetische Operationen verknüpfen den Inhalt von AKKU 1 und AKKU 2, schreiben das Ergebnis in den AKKU 1 und übertragen den Inhalt des AKKU 3 nach AKKU 2 und den Inhalt des AKKU 4 nach AKKU 3 (Stack Drop); bei 16-bit-Festpunktarithmetik werden nur das Low-Word von AKKU 3 in das Low-Word von AKKU 2 und das Low-Word von AKKU 4 in das Low-Word von AKKU 3 übertragen,

- beim Addieren einer Konstanten (ADD BF/KF/DH) zum Inhalt des AKKU 1 werden die Akkumulatoren 2, 3 und 4 nicht verändert.

Ergebnisanzeigen

STEP-5-Operationen sind entweder anzeigenbildend oder anzeigenabfragend. Die Anzeigen werden in einem Anzeigenbyte hinterlegt. Bei den Anzeigen sind zwei Gruppen zu unterscheiden: Anzeigen von digitalen Operationen (Wortanzeigen – Bit 4 bis 7 im Anzeigenbyte) und Anzeigen von binären und organisatorischen Operationen (Bit-Anzeigen – Bit 0 bis 3 im Anzeigenbyte). Wie die verschiedenen Anzeigen durch STEP-5-Operationen beeinflußt bzw. ausgewertet werden können, entnehmen Sie bitte der Operationsliste. /1/.

Das Anzeigenbyte wird bei der PG-Online-Funktion "Status Baustein" (siehe Abschnitt 11.2.3) angezeigt und hat folgenden Aufbau:

Wort-Anzeigen				**Bit-Anzeigen**			
ANZ 1	ANZ 0	OV	OS	OR	STA	VKE	$\overline{\text{ERAB}}$
Bit 7	6	5	4	3	2	1	0

Bit-Anzeigen

- **$\overline{\text{ERAB}}$** Erstabfrage

  Eine logische Verknüpfungskette mit binären Verknüpfungen **beginnt** immer mit einer sogenannten **Erstabfrage**, bei der das VKE **neu** gebildet wird. Mit ihr wird die Bit-Anzeige ERAB = 1 gesetzt. Im Laufe weiterer logischer Verknüpfungen in der begonnenen Kette bleibt ERAB = 1, und das VKE kann durch diese logischen Verknüpfungen verändert werden.

  Die begonnene Verknüpfungskette wird **beendet** durch eine binäre Speicheroperation (z. B. S A 5.0). Mit der Speicheroperation wird ERAB = 0 gesetzt; das VKE kann jetzt nur noch ausgewertet (z. B. durch VKE-abhängige Befehle), aber nicht mehr weiter verknüpft werden. Die nächste binäre logische Verknüpfung nach einer binären Speicheroperation ist wieder eine "Erstabfrage".

Beispiel zu $\overline{ERAB}$

```
:S    A  7.7     letzte Operation der vorher-
:                gehenden Verknüpfungskette
:U    E  1.0     ERAB wird auf '1' gesetzt,
:                VKE wird durch UND-Ver-
:                knüpfung neu gebildet
:O    E  6.3     VKE wird durch ODER-Verknüp-
:                fung beeinflußt
:UN   E  2.1     VKE wird durch UND-NICHT-
:                Verknüpfung beeinflußt
:S    A  2.4     ERAB wird auf '0' gesetzt,
:                Verknüpfungskette ist beendet
:SPB  FB 150     Funktionsbaustein in Abhängig-
:                keit vom VKE aufrufen
:
```


weitere Bit-Anzeigen

- **VKE** Verknüpfungsergebnis

  Ergebnis binärer Verknüpfungen. Wahrheitsaussage bei den Vergleichsbefehlen (siehe Operationsliste, binäre Verknüpfungsoperationen bzw. Vergleichsoperationen).

- **STA** Status

  Gibt bei Bit-Befehlen den logischen Zustand des gerade abgefragten oder gesetzten Bits an. Der Status wird bei binären Verknüpfungsoperationen – ausgenommen U(, O(,), O – und bei Speicheroperationen aktualisiert.

- **OR** Oder

  Interne Anzeige der CPU für die Behandlung von "UND-vor-ODER-Verknüpfungen".

Wort-Anzeigen

- **OV** Overflow

  Gibt an, ob bei der eben abgeschlossenen arithmetischen Operation der zulässige Zahlenbereich überschritten worden ist.

- **OS** Overflow speichernd

  OS dient dazu, im Verlaufe mehrerer arithmetischer Operationen zu erkennen, ob irgendwann ein Überlauf (Overflow) aufgetreten ist.

- **ANZ 1 und ANZ 0**

  Codierte Ergebnisanzeigen, deren Interpretation aus der folgenden Tabelle ersichtlich wird.

Hinweis
Zur unmittelbaren Auswertung der Anzeigen stehen Vergleichs- und Sprungoperationen zur Verfügung (siehe Abschnitte 3.5.1 und 3.5.3).

Tabelle 3-1 Ergebnisanzeigen von STEP-5-Operationen

Wort-Anzeigen		Arithmetische Operationen	Digitale Verknüpfungsoperationen	Vergleichsoperationen	Schiebeoperationen	Bei SES, SEF	Ausgeführte Sprungoperationen
ANZ 1	ANZ 0						
0	0	Ergebnis = 0	Ergebnis = 0	AKKU 2 = AKKU 1	geschobenes Bit = 0	Semaphor ist gesetzt	SPZ
0	1	Ergebnis < 0	–	AKKU 2 < AKKU 1	–	–	SPM SPN
1	0	Ergebnis > 0	Ergebnis ≠ 0	AKKU 2 > AKKU 1	geschobenes Bit = 1	Semaphor wird gesetzt bzw. freigegeben	SPP SPN
1	1	Division durch Null	–	–	–	–	SPN

Hinweis
Bei einem Ebenenwechsel, z. B. bei der Bearbeitung eines Weckalarms, werden alle Akkumulatoren sowie die Bit- und Wortanzeigen (VKE usw.) gerettet und bei Fortsetzung der unterbrochenen Ebene wieder geladen.

3.5.1 Grundoperationen

Die Grundoperationen können Sie in **allen** Codebausteinen und in allen Darstellungsarten (KOP, FUP und AWL) benutzen

Binäre Verknüpfungsoperationen

Tabelle 3-2 Binäre Verknüpfungsoperationen

Operation	Operand	Funktion
U		UND-Verknüpfung mit Abfrage auf Signalzustand "1"
O		ODER-Verknüpfung mit Abfrage auf Signalzustand "1"
	E 0.0 bis 127.7	eines Eingangs im PAE
	A 0.0 bis 127.7	eines Ausgangs im PAA
	M 0.0 bis 255.7	eines Merkerbits
	S 0.0 bis 1023.7	eines S-Merkerbits
	D 0.0 bis 255.15	eines Datenwortbits
	T 0 bis 255	einer Zeit
	Z 0 bis 255	eines Zählers
UN		UND-Verknüpfung mit Abfrage auf Signalzustand "0"
ON		ODER-Verknüpfung mit Abfrage auf Signalzustand "0"
	E 0.0 bis 127.7	eines Eingangs im PAE
	A 0.0 bis 127.7	eines Ausgangs im PAA
	M 0.0 bis 255.7	eines Merkerbits
	S 0.0 bis 1023.7	eines S-Merkerbits
	D 0.0 bis 255.15	eines Datenwortbits
	T 0 bis 255	einer Zeit
	Z 0 bis 255	eines Zählers
O	–	ODER-Verknüpfung von UND-Funktionen
U(	–	UND-Verknüpfung von Klammerausdrücken
O(		ODER-Verknüpfungen von Klammerausdrücken
)		Klammer zu (Abschluß eines Klammerausdrucks)
		Es sind maximal 8 Ebenen zulässig, d. h. 7 geöffnete Klammern.

VKE-Bildung

Die binären Verknüpfungsoperationen erzeugen das Verknüpfungsergebnis (VKE).
Am Anfang einer Verknüpfungskette hängt die Bildung des VKE (Erstabfrage) nur vom abgefragten Signalzustand (Status) ab, jedoch nicht von der Verknüpfungsart (O = ODER, U = UND).

Innerhalb einer Verknüpfungskette wird das VKE aus Verknüpfungsart, bisherigem VKE und dem abgefragten Signalzustand gebildet. Eine Verknüpfungskette wird durch einen VKE-begrenzenden ($\overline{ERAB} = 0$) Befehl (z. B. Speicheroperationen) abgeschlossen. Danach kann das VKE zwar ausgewertet, jedoch nicht weiter verknüpft werden.

Beispiel

Programm	STA	VKE	$\overline{ERAB}$
:			
= A 0.0	0	0	0 ← VKE-begrenzt
U E 1.0	1 →	1	1 ← Erstabfrage
U E 1.1	1 →	1	1
U E 1.2	0 →	0	1
= A 0.1	0	0	0 ← VKE-begrenzt, Ende der Verknüpfungskette

Speicheroperationen

Tabelle 3-3 Speicheroperationen

Operation	Operand	Funktion
S		Bei Zustand ’1’ des VKE: Setzen
R		Bei Zustand ’1’ des VKE: Zurücksetzen
	E 0.0 bis 127.7	eines Eingangs im PAE
	A 0.0 bis 127.7	eines Ausgangs im PAA
	M 0.0 bis 255.7	eines Merkers
	S 0.0 bis 1023.7	eines S-Merkers
	D 0.0 bis 255.15	eines Bits im Datenwort
=		Zuweisung des VKE zu einem
	E 0.0 bis 127.7	Eingang im PAE
	A 0.0 bis 127.7	Ausgang im PAA
	M 0.0 bis 255.7	Merker
	S 0.0 bis 1023.7	S-Merker
	D 0.0 bis 255.15	Bit im Datenwort

Lade- und Transferoperationen

Tabelle 3-4 Lade- und Transferoperationen/Teil 1

Operation	Operand	Funktion
L T		Laden Transferieren
	EB 0 bis 127	eines Eingabebytes vom/zum PAE
	EW 0 bis 126	eines Eingabewortes vom/zum PAE
	ED 0 bis 124	eines Eingabe-Doppelwortes vom/zum PAE
	AB 0 bis 127	eines Ausgabebytes vom/zum PAA
	AW 0 bis 126	eines Ausgabewortes vom/zum PAA
	AD 0 bis 124	eines Ausgabe-Doppelwortes vom/zum PAA
	MB 0 bis 255	eines Merkerbytes
	MW 0 bis 254	eines Merkerwortes
	MD 0 bis 252	eines Merker-Doppelwortes
	SY 0 bis 1023	eines S-Merkerbytes
	SW 0 bis 1022	eines S-Merkerwortes
	SD 0 bis 1020	eines S-Merker-Doppelwortes
	DR 0 bis 255	des rechten Bytes eines Datenwortes aus bzw. nach DB, DX
	DL 0 bis 255	des linken Bytes eines Datenwortes aus bzw. nach DB, DX
	DW 0 bis 255	eines Datenwortes aus bzw. nach DB, DX
	DD 0 bis 254	eines Daten-Doppelwortes aus bzw. nach DB, DX
	PY 0 bis 127	eines Peripheriebytes der Digitaleingaben bzw. -ausgaben (P-Bereich)
	PY 128 bis 255	eines Peripheriebytes der Analog- oder Digitaleingaben bzw. -ausgaben (P-Bereich)
	PW 0 bis 126	eines Peripheriewortes der Digitaleingaben bzw. -ausgaben (P-Bereich)
	PW 128 bis 254	eines Peripheriewortes der Analog- oder Digitaleingaben bzw. -ausgaben (P-Bereich)
	QB 0 bis 255	eines Bytes der erweiterten Peripherie (Q-Bereich)
	QW 0 bis 254	eines Wortes der erweiterten Peripherie (Q-Bereich)

Tabelle 3-5 Lade- und Transferoperationen/Teil 2

Operation	Operand		Funktion
L			Laden
	KB	0 bis 255	einer Konstanten, Bytewert
	KC	2 ASCII-Zeichen	einer Konstanten als 2 ASCII-Zeichen
	KF	-32768 bis +32767	einer Konstanten als Festpunktzahl
	KG	[1]	einer Konstanten als Gleitpunktzahl
	KH	0 bis FFFF	einer Konstanten als Hexadezimalzahl
	DH	0 bis FFFF FFFF	einer Doppelwort-Konstanten als Hexadezimalzahl
	KM	16-bit-Muster	einer Konstanten als Bitmuster
	KY	0 bis 255 für jedes Byte	einer Konstanten als 2-Byte-Zahl
	KT	0.0 bis 999.3	einer Konstanten als Zeitwert (BCD-codiert)
	KZ	0 bis 999	einer Konstanten als Zählwert
	T	0 bis 255	eines Zeitwertes, dual-codiert
	Z	0 bis 255	eines Zählwertes, dual-codiert
LC			Laden eines
	T	0 bis 255	Zeitwertes
	Z	0 bis 255	Zählwertes
			im BCD-Code

[1] $\pm 0{,}1469368 \times 10^{-38}$ bis $\pm 0{,}1701412 \times 10^{39}$

Ladeoperationen

Lade operationen schreiben den adressierten Wert in den AKKU 1, dessen vorheriger Inhalt in den AKKU 2 gerettet wird (Stack Lift).

Transferoperationen

Transferoperationen schreiben den Inhalt des AKKU 1 in die adressierte Speicherzelle.

Beispiele zu den Lade- und Transferoperationen

Beispiel 1:

Bild 3-6 zeigt das Laden/Transferieren eines Bytes, Wortes oder Doppelwortes aus einem/in einen **byteweise** organisierten Speicherbereich (PAE, PAA, Merker, Peripherie):

```
:L EB i geladen wird Byte i des PAE in den AKKU-1-LL
:L EW j geladen werden Byte j und j+1 des PAE in den AKKU-1-L
:L MD k geladen werden die Merkerbytes k bis k+3 in den AKKU 1
```

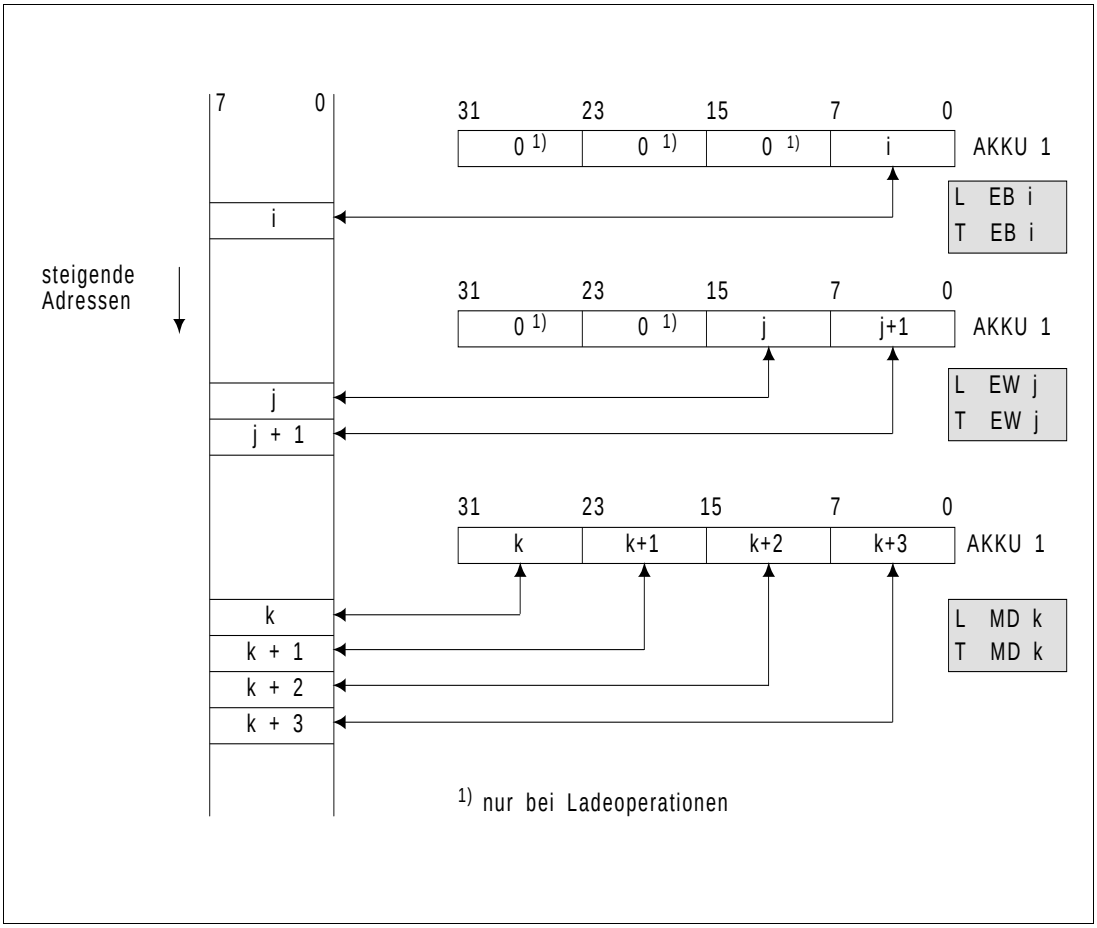
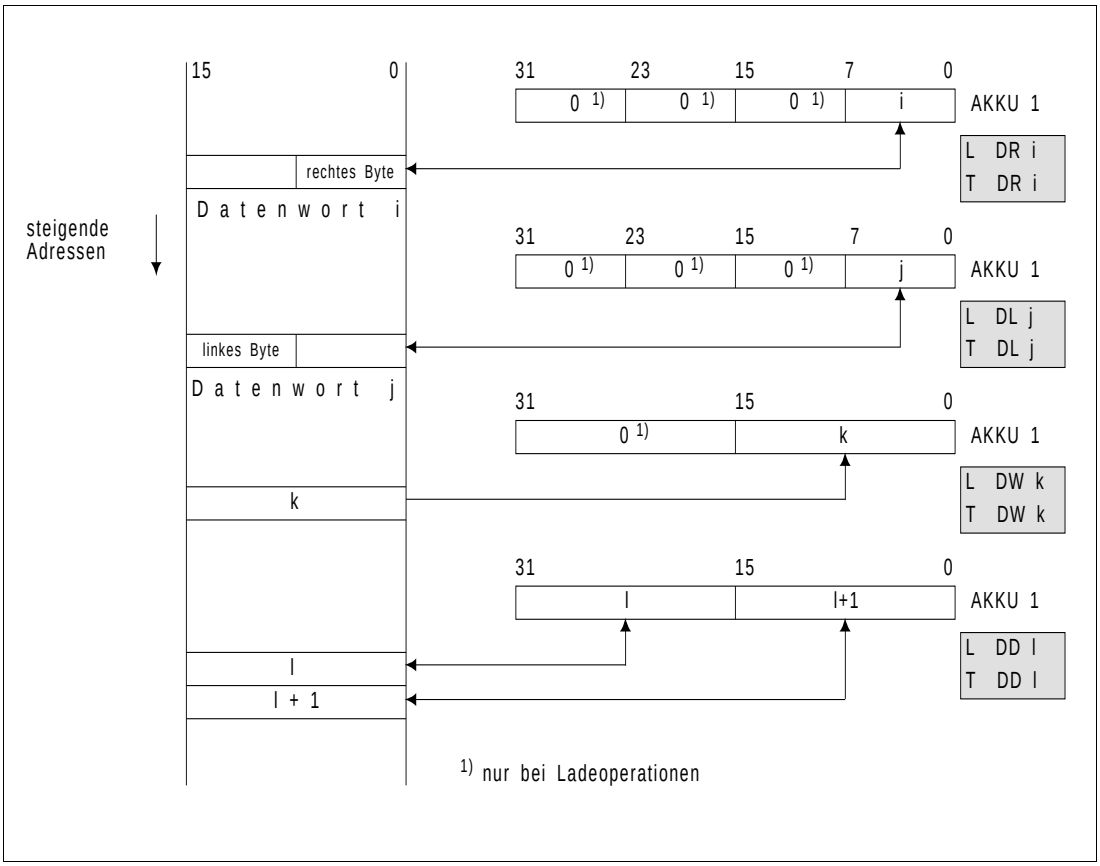


Bild 3-6 Lade- und Transferoperationen in einem byteweise orientierten Speicherbereich

Beispiel 2:

Bild 3-7 zeigt das Laden/Transferieren eines Bytes, Wortes oder Doppelwortes aus einem/in einen **wortweise** organisierten Speicherbereich.

```
:L DR i geladen wird das rechte Byte aus dem Datenwort i in den AKKU-1-LL
:L DL j geladen wird das linke Byte aus dem Datenwort  j in den AKKU-1-LL
:L DW k geladen wird das Datenwort k in den AKKU-1-L
:L DD l geladen werden die Datenwörter l und l+1 in den AKKU 1
```



Hinweis
Ladeoperationen beeinflussen die **Anzeigen nicht**.
Transferoperationen löschen das **OS-Bit**.

Beim **Laden** eines **Bytes** oder **Wortes** werden die **höherwertigen Bytes** im AKKU **1 gelöscht.**

Ansprechen der Peripherie

Die Peripherie kann durch Lade- und Transferoperationen angesprochen werden:

- **direkt:**

  mit L../T.. ..PY, ..PW, ..QB, ..QW

oder

- **über das Prozeßabbild:**

  mit L../T.. ..EB, ..EW, ..ED, .AB, ..AW, ..AD

  und mit Verknüpfungsoperationen.

Hinweis
Bei den Transferoperationen T PY 0 bis 127 und T PW 0 bis 126 wird parallel das Prozeßabbild der Ausgänge nachgeführt.

Beachten Sie zur Peripherie folgende Punkte:

- Ein Prozeßabbild der Ein- und Ausgänge existiert für je 128 Ein- und Ausgabebytes der P-Peripherie mit Byteadressen von 0 bis 127.
- Für den Bereich der P-Peripherie mit den relativen Byteadressen von 128 bis 255 und den gesamten Bereich der Q-Peripherie existiert kein Prozeßabbild! (Zur Adreßraumaufteilung der Peripherie siehe Abschnitt 8.2.2.)
- Ein-/Ausgabebaugruppen mit Adressen der Q-Peripherie können nur in Erweiterungsgeräten stecken (nicht im Zentralgerät).
- In **einem** Erweiterungsgerät kann man entweder nur P-Peripherie oder nur Q-Peripherie verwenden.

Vorsicht
Falls in einem Erweiterungsgerät Relativadressen der Q-Peripherie verwendet werden, sind diese Adressen für Peripheriebaugruppen (P-Bereich) im Zentralgerät nicht mehr zulässig (Doppeladressierung!).

Zeit- und Zähloperationen

Um eine Zeit durch eine Startoperation oder einen Zähler durch eine Setzoperation zu laden, muß der Wert vorher in den AKKU 1 geladen werden.

Zu bevorzugen sind folgende Ladeoperationen:

Für Zeiten: L KT, L EW, L AW, L MW, L DW, L SW.
Für Zähler: L KZ, L EW, L AW, L MW, L DW, L SW.

Damit eine **Zeit** mit dem vorgegebenen Zeitwert gestartet wird, ist ein Flankenwechsel des VKE notwendig.

Ein **Zähler** wird mit dem vorgegebenen Zählwert gesetzt oder gezählt, wenn eine positive Flanke des VKE erkannt wird.

In der nachfolgenden Tabelle sind die Flankenwechsel in der Spalte VKE mit entsprechenden Pfeilen gekennzeichnet.

Tabelle 3-6 Zeit- und Zähloperationen

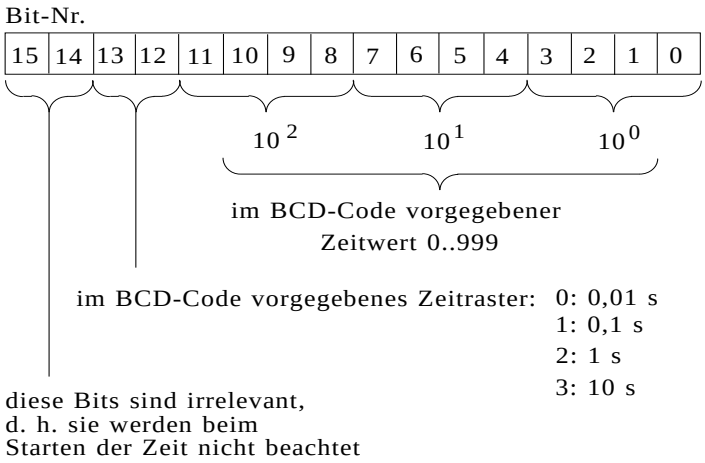
Operation	**Operand**	**VKE 1)**	**Funktion**
SI	T 0 bis 255	↑	Starten einer Zeit als Impuls
SV	T 0 bis 255	↑	Starten einer Zeit als verlängerter Impuls
SE	T 0 bis 255	↑	Starten einer Zeit als Einschaltverzögerung
SS	T 0 bis 255	↑	Starten einer Zeit als speichernde Einschaltverzögerung
SA	T 0 bis 255	↓	Starten einer Zeit als Ausschaltverzögerung
R	T 0 bis 255	1	Rücksetzen einer Zeit
S	Z 0 bis 255	↑	Setzen eines Zählers (BCD-Wert von 0 bis 999)
R	Z 0 bis 255	1	Rücksetzen eines Zählers
ZV	Z 0 bis 255	↑	Vorwärtszählen eines Zählers
ZR	Z 0 bis 255	↑	Rückwärtszählen eines Zählers

1) positive Flanke (↑): Signalzustandsänderung von Zustand ’0’ nach Zustand ’1’
negative Flanke (↓): Signalzustandsänderung von Zustand ’1’ nach Zustand ’0’

Bei der Ausführung der Zeit- bzw. Zähloperationen SI, SE, SV, SS, SA und S wird der im AKKU 1 stehende Wert in die Zeit- bzw. Zählzelle gebracht (entspricht dem Transferbefehl) und die entsprechende Operation veranlaßt.

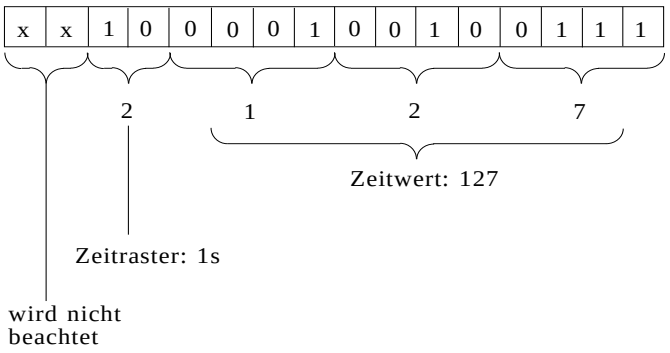
Zeitwert

Ein **Zeitwert** kann mit der Operation L KT direkt oder mit entsprechenden Ladeoperationen aus einem Merker- oder Datenwort indirekt in den AKKU 1 geladen werden. Er muß folgenden Aufbau haben (das Zeitraster wird bei L KT hinter dem Punkt im Operanden angegeben):



Beispiel

```
Es soll eine Zeitdauer von 127 s vorgegeben werden:

Belegung der Bits:
```



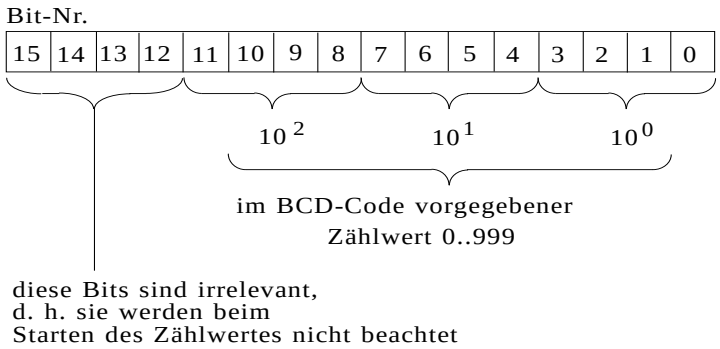
Hinweis
Jedes Starten einer Zeit ist mit der Ungenauigkeit von 1 Zeitraster verbunden! Wählen Sie deshalb bei Verwendung von Zeitzellen ein möglichst kleines Zeitraster.

Beispiel:

Zeitdauer 4s	nicht :	1 s x 4	Ungenauigkeit: 1 s
	sondern:	0,01 s x 400	Ungenauigkeit: 0,01 s

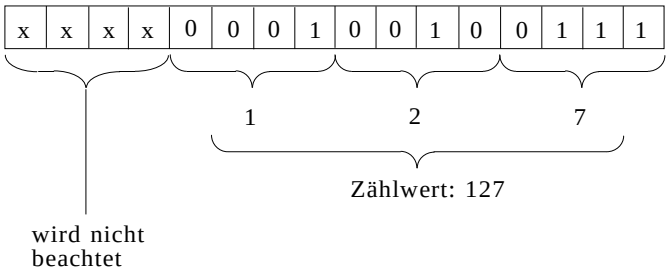
Zählwerte

Ein **Zählwert** kann mit dem Befehl L KZ direkt oder mit entsprechenden Ladeoperationen aus einem Merker- oder Datenwort indirekt in den AKKU 1 geladen werden. Er muß folgenden Aufbau haben:



Beispiel

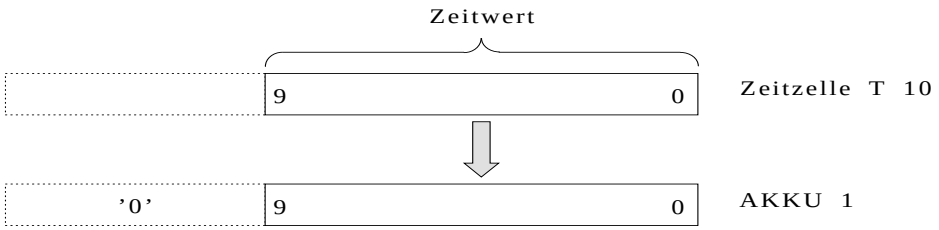
```
Es soll ein Zählwert von 127 vorgegeben werden:

Belegung der Bits:
```



In der Zeit- bzw. Zählzelle selbst liegt der Zeit- bzw. Zählwert dualcodiert vor. Zur Abfrage der Zeit bzw. des Zählers kann der Wert der Zeit- bzw. Zählzelle **direkt** oder **BCD-codiert** in den AKKU 1 geladen werden.

Weitere Beispiele zu Zeit- und Zählwerten

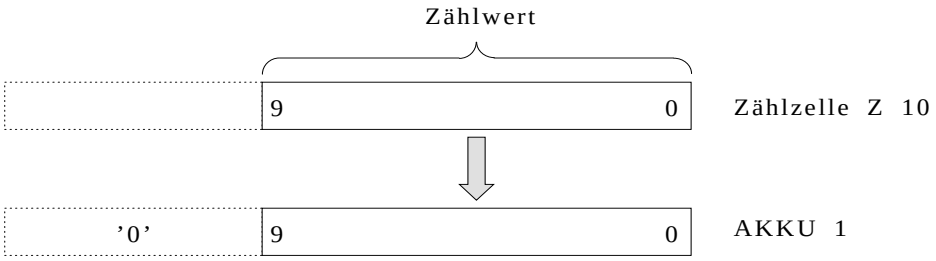
Direktes Laden von **Zeitwerten**:



"L T 10": Direktes Laden des dualen Zeitwertes der Zeit T 10
in den AKKU 1

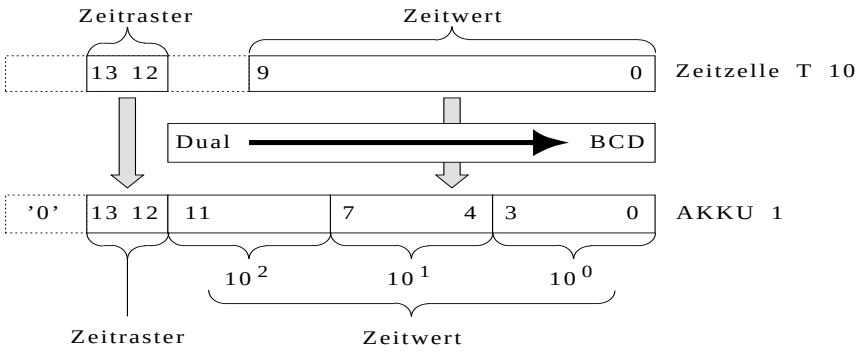
Das Zeitraster wird nicht mitgeladen.

Direktes Laden von **Zählwerten:**



"L Z 10": Direktes Laden des dualen Zählwertes des Zählers Z 10
in den AKKU 1

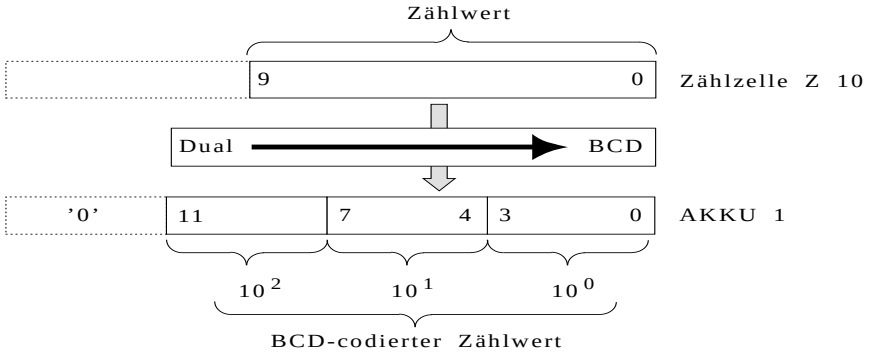
Codiertes Laden von **Zeitwerten:**



"LC T 10": Codiertes Laden des Zeitwertes und des Zeitrasters der Zeit T 10 in den AKKU 1

Das Zeitraster wird mitgeladen.

Codiertes Laden von **Zählwerten:**



"LC Z 10": Codiertes Laden des Zählwertes des Zählers Z 10 in den AKKU 1

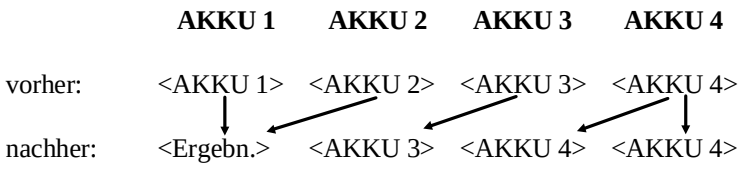
Beim codierten Laden werden die Zustandsbits 14 und 15 der Zeitzellen bzw. 12 bis 15 der Zählzellen nicht geladen. An ihrer Stelle steht 0 in AKKU 1. Der nun im Akku stehende Wert kann weiterverarbeitet werden.

Arithmetische Operationen

Tabelle 3-7 Arithmetische Operationen

Operation	Operand	Funktion
+ F	–	Addition zweier Festpunktzahlen (16 bit)
- F		Subtraktion zweier Festpunktzahlen (16 bit)
x F		Multiplikation zweier Festpunktzahlen (16 bit)
: F		Division zweier Festpunktzahlen (16 bit): Quotient in AKKU-1-L, Rest in AKKU-1-H
+ G		Addition zweier Gleitpunktzahlen (32 bit)
- G		Subtraktion zweier Gleitpunktzahlen (32 bit)
x G		Multiplikation zweier Gleitpunktzahlen (32 bit)
: G		Division zweier Gleitpunktzahlen (32 bit)

Die arithmetischen Operationen verknüpfen den Inhalt von AKKU 1 und AKKU 2 (z. B. "AKKU 2 - AKKU 1"). Das Ergebnis steht anschließend im AKKU 1. Die Rechenregister werden durch eine arithmetische Operation wie folgt verändert (bei Festpunkt-Operationen nur das Low-Word):



Hinweis
Innerhalb der **Ergänzenden Operationen** stehen Ihnen Operationen zur **Subtraktion** und **Addition** von **Doppelwort-Festpunktzahlen** zur Verfügung.
Ferner können Sie von den Ergänzenden Operationen die Operation **ENT** für das Laden von AKKU 3 und AKKU 4 benutzen (siehe Abschnitt 3.5.4).

Vergleichsoperationen

Tabelle 3-8 Vergleichsoperationen

Operation	Operand	Funktion
! = > > > = < <= F D G	–	Vergleich auf gleich Vergleich auf ungleich Vergleich auf größer Vergleich auf größer/gleich Vergleich auf kleiner Vergleich auf kleiner/gleich ...F: Vergleich zweier Festpunktzahlen (16 bit) ...D: Vergleich zweier Festpunktzahlen (32 bit) ...G: Vergleich zweier Gleitpunktzahlen (32 bit)

Bausteinoperationen

Tabelle 3-9 Bausteinoperationen

Operation	Operand	Funktion
S P A S P B	 OB 1 bis 39 [1)] OB 110 bis 255 PB 0 bis 255 FB 0 bis 255 SB 0 bis 255	Sprung unbedingt Sprung bedingt (nur wenn VKE = 1) zu einem Organisationsbaustein zu einer Sonderfunktion zu einem Programmbaustein zu einem FB-Funktionsbaustein zu einem Schrittbaustein
B A B A B	 FX 0 bis 255	Sprung unbedingt Sprung bedingt (nur wenn VKE = 1) zu einem FX-Funktionsbaustein
B E B E B B E A	–	Bausteinende Bausteinende bedingt (nur wenn VKE = 1) Bausteinende absolut
A A X	DB 3 bis 255 DX 3 bis 255	Aufschlagen eines DB-Datenbausteins Aufschlagen eines DX-Datenbausteins
E EX	DB 3 bis 255 DX 3 bis 255	Datenbaustein DB erzeugen Datenbaustein DX erzeugen (AKKU 1 muß die Anzahl der Datenwörter – max. 4091– enthalten, die der neue Baustein haben soll)

[1)] nur für Testzwecke!

E DB/EX DX

Erzeuge Datenbaustein

Die Operation E DBx erzeugt einen DB-Datenbaustein mit der Nummer x ($3 \leq x \leq 255$) im Datenbaustein-RAM der CPU. Der Inhalt des Datenbausteins wird dabei **nicht** mit Null besetzt, d.h. die Datenwörter haben beliebige Inhalte.
Vor dem Programmieren der Anweisung müssen Sie die Anzahl der Datenwörter, die der neue DB haben soll, im AKKU-1-L hinterlegen. Der dazugehörige Bausteinkopf wird von der Operation "E DB" bzw. "EX DX" erzeugt. Ein so erzeugter Datenbaustein darf (**ohne** Bausteinkopf) maximal 4091 Wörter lang sein.

Falls der entsprechende Datenbaustein schon existiert, die Länge des DB unzulässig ist oder der Platz im DB-RAM nicht ausreicht, ruft das Systemprogramm den **OB 31** auf. Wenn dieser nicht geladen ist, geht die CPU aufgrund eines Laufzeitfehlers in den Stoppzustand.

Die Operation EX DX erzeugt im DB-RAM einen DX-Datenbaustein und arbeitet wie E DB.

Null-/Bildaufbau-/Stopp-Operationen

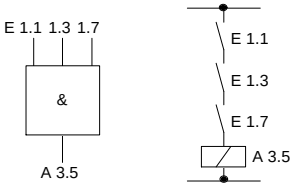
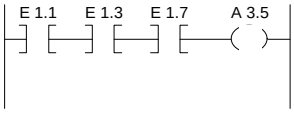
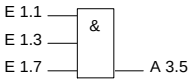
Tabelle 3-10 Null-/Bildaufbau-/Stopp-Operationen

Operation	Operand	Funktion
N O P 0 N O P 1	–	Nulloperation Nulloperation
B L D	0 bis 255	Bildaufbauanweisung für das PG: wird von der CPU wie eine Nulloperation behandelt
S T P	–	CPU geht den STOP.

3.5.2 Programmierbeispiele in den Darstellungsarten AWL, KOP und FUP

Verknüpfungsoperationen

UND-Verknüpfung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	U E 1.1 U E 1.3 U E 1.7 = A 3.5		

Am Ausgang A 3.5 erscheint Signalzustand "1", wenn alle Eingänge gleichzeitig den Signalzustand "1" aufweisen.

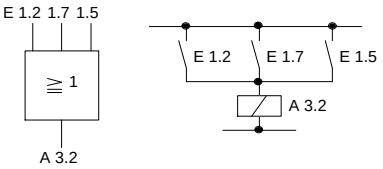
Am Ausgang A 3.5 erscheint Signalzustand "0", wenn mindestens einer der Eingänge den Signalzustand "0" aufweist.

Die Anzahl der Abfragen und die Reihenfolge der Programmierung ist beliebig.

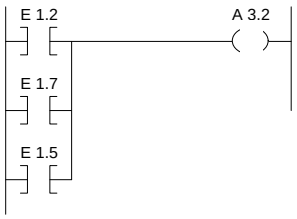
Verknüpfungsoperationen
(Fortsetzung)

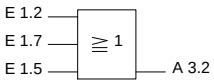
ODER-Verknüpfung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
O   E 1.2
O   E 1.7
O   E 1.5
=   A 3.2
```





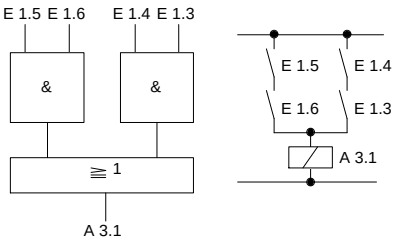
Am Ausgang A 3.2 erscheint Signalzustand "1", wenn mindestens einer der Eingänge den Signalzustand "1" aufweist.

Am Ausgang A 3.2 erscheint Signalzustand "0", wenn alle Eingänge gleichzeitig den Signalzustand "0" aufweisen.

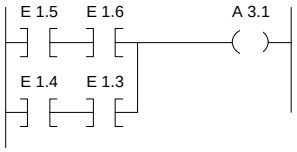
Die Anzahl der Abfragen und die Reihenfolge der Programmierung ist beliebig.

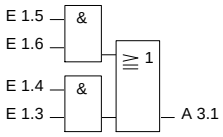
UND-vor-ODER-Verknüpfung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U   E 1.5
U   E 1.6
O
U   E 1.4
U   E 1.3
=   A 3.1
```



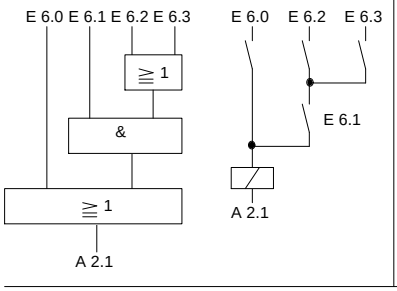
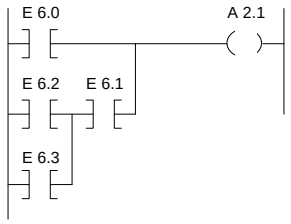
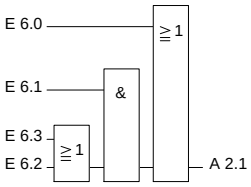


Am Ausgang A 3.1 erscheint Signalzustand "1", wenn mindestens eine UND-Verknüpfung erfüllt ist.

Am Ausgang A 3.1 erscheint Signalzustand "0", wenn keine UND-Verknüpfung erfüllt ist.

Verknüpfungsoperationen
(Fortsetzung)

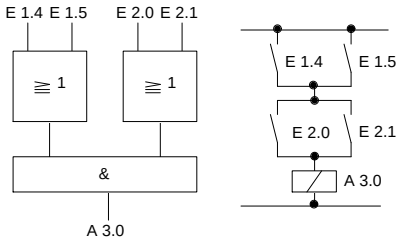
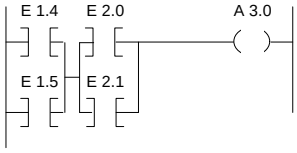
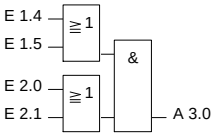
ODER-vor-UND-Verknüpfung 1. Beispiel

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	U E 6.0 O U E 6.1 U (O E 6.2 O E 6.3) = A 2.1		

Am Ausgang A 2.1 erscheint Signalzustand "1", wenn Eingang E 6.0 oder Eingang E 6.1 und einer der Eingänge E 6.2 bzw. E 6.3 Signal "1" führen.

Am Ausgang A 2.1 erscheint Signalzustand "0", wenn Eingang E 6.0 Signal "0" führt und die UND-Verknüpfung nicht erfüllt ist.

ODER-vor-UND-Verknüpfung 2. Beispiel

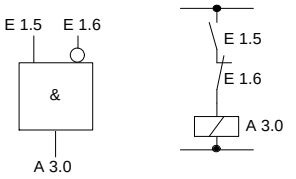
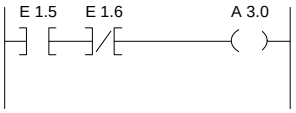
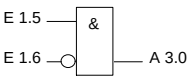
Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	U (O E 1.4 O E 1.5) U (O E 2.0 O E 2.1) = A 3.0		

Am Ausgang A 3.0 erscheint Signalzustand "1", wenn beide ODER-Verknüpfungen erfüllt sind.

Am Ausgang A 3.0 erscheint Signalzustand "0", wenn mindestens eine ODER-Verknüpfung nicht erfüllt ist.

Verknüpfungsoperationen
(Fortsetzung)

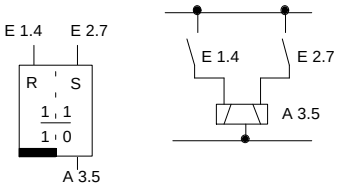
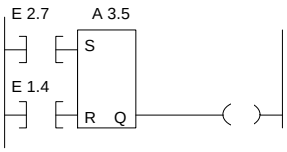
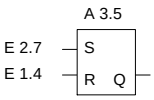
Abfrage auf Signalzustand "0"

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	U E 1.5 UN E 1.6 = A 3.0		

Am Ausgang A 3.0 erscheint Signalzustand "1" nur dann, wenn der Eingang E 1.5 den Signalzustand "1" (Schließer betätigt) und der Eingang E 1.6 den Signalzustand "0" (Öffner nicht betätigt) führt.

Speicheroperationen

RS-Speicherglied für speichernde Signalausgabe

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	U E 2.7 S A 3.5 U E 1.4 R A 3.5		

Signalzustand "1" am Eingang E 2.7 bewirkt das Setzen des Speicherglieds (Signalzustand "1" am Ausgang A 3.5). Wechselt der Signalzustand am Eingang E 2.7 nach "0", so bleibt dieser Zustand erhalten, d. h., das Signal wird gespeichert.

Signalzustand "1" am Eingang E 1.4 bewirkt das Rücksetzen des Speichergliedes (Signalzustand "0" am Ausgang A 3.5).

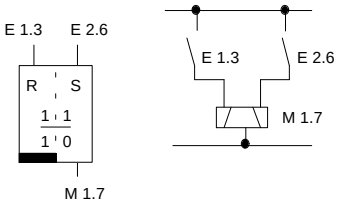
Wechselt der Signalzustand am Eingang E 1.4 nach "0", so bleibt dieser Zustand erhalten.

Bei gleichzeitigem Anliegen des Setzsignals (Eingang E 2.7) und des Rücksetzsignals (Eingang E 1.4) ist die zuletzt programmierte Abfrage (hier U E 1.4) während der Bearbeitung des übrigen Programms wirksam (Rücksetzen vorrangig).

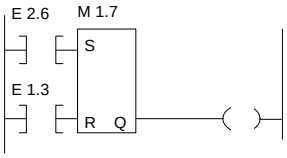
Speicheroperationen
(Fortsetzung)

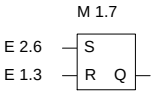
RS-Speicherglied mit Merkern

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U   E  2.6
S   M 1.7
U   E  1.3
R   M 1.7
```





Signalzustand "1" am Eingang E 2.6 bewirkt das Setzen des Speichergliedes.

Wechselt der Signalzustand am Eingang E 2.6 nach "0", so bleibt dieser Zustand erhalten, d. h. das Signal wird gespeichert.

Signalzustand "1" am Eingang E 1.3 bewirkt das Rücksetzen des Speichergliedes.

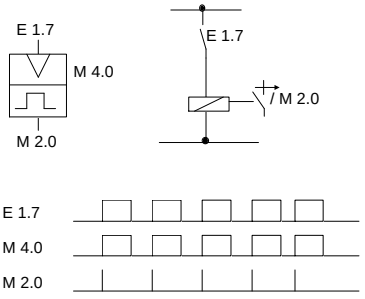
Wechselt der Signalzustand am Eingang E 1.3 nach "0", so bleibt dieser Zustand erhalten.

Bei gleichzeitigem Anliegen des Setzsignals (Eingang E 2.6) und des Rücksetzsignals (Eingang E 1.3) ist die zuletzt programmierte Abfrage (hier U E 1.3) während der Bearbeitung des übrigen Programms wirksam (Rücksetzen vorrangig).

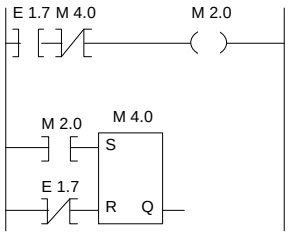
Speicheroperationen
(Fortsetzung)

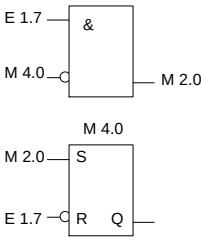
Nachbildung eines Wischrelais

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungsliste	Kontaktplan	Funktionsplan




```
U   E 1.7
UN  M 4.0
=   M 2.0
U   M 2.0
S   M 4.0
UN  E 1.7
R   M 4.0
```





Bei jeder ansteigenden Flanke des Eingangs E 1.7 ist die UND-Verknüpfung (U E 1.7 und UN M 4.0) erfüllt und mit VKE = "1" werden die Merker M 4.0 ("Flankenmerker") und M 2.0 ("Impulsmerker") gesetzt.

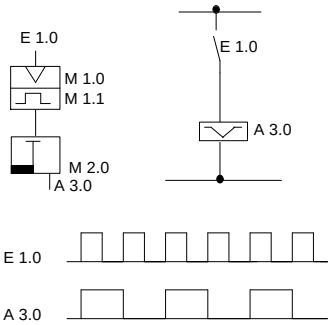
Beim nächsten Bearbeitungszyklus ist die UND-Verknüpfung U E 1.7 und UN M 4.0 nicht erfüllt, da der Merker M 4.0 gesetzt worden ist.

Der Merker M 2.0 wird rückgesetzt.

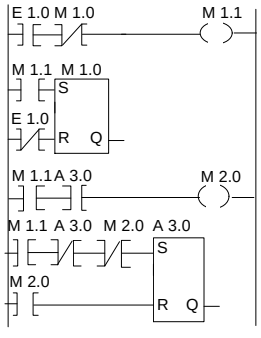
Der Merker M 2.0 führt also während eines einzigen Programmdurchlaufs Signalzustand "1".

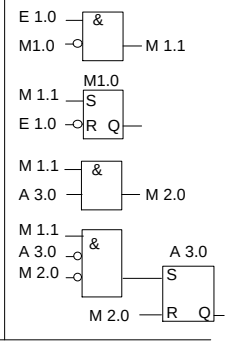
Binäruntersetzer (T-Kippglied)

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungsliste	Kontaktplan	Funktionsplan




```
U   E 1.0
UN  M 1.0
=   M 1.1
U   M 1.1
S   M 1.0
UN  E 1.0
R   M 1.0
U   M 1.1
U   A 3.0
=   M 2.0
U   M 1.1
UN  A 3.0
UN  M 2.0
S   A 3.0
U   M 2.0
R   A 3.0
```



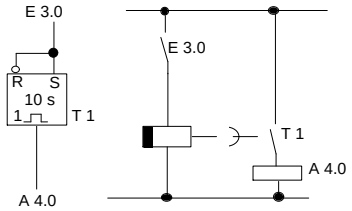


Der Binäruntersetzer (Ausgang A 3.0) wechselt bei jedem Signalzustandswechsel von "0" nach "1" (ansteigende Flanke) des Einganges E 1.0 seinen Zustand. Am Ausgang des Speicherglieds erscheint deshalb die halbe Eingangsfrequenz.

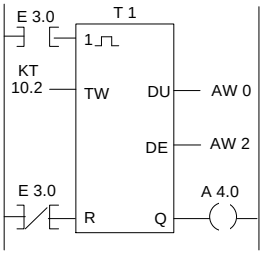
Zeitoperationen

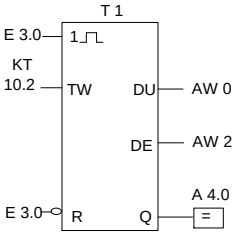
Impuls

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U    E   3.0
L    KT 10.2
SI   T   1
UN   E   3.0
R    T   1
L    T   1
T    AW  0
LC   T   1
T    AW  2
U    T   1
=    A   4.0
```





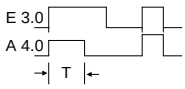
Bei Signalzustandswechsel von "0" nach "1" am Eingang E 3.0 wird das Zeitglied gestartet. Bei wiederholter Bearbeitung mit Verknüpfungsergebnis "1" bleibt das Zeitglied unbeeinflußt.

Bei Signalzustand "0" am Eingang E 3.0 wird das Zeitglied auf Null gesetzt (gelöscht).

Die Abfragen U T bzw. O T liefern Signalzustand "1", solange die Zeit läuft.

KT 10.2:

Das Zeitglied wird mit dem angegebenen Wert (10) geladen.
Die Zahl rechts vom Punkt gibt das Zeitraster an:
0 = 001 s 2 = 1 s
1 = 0.1 s 3 = 10 s

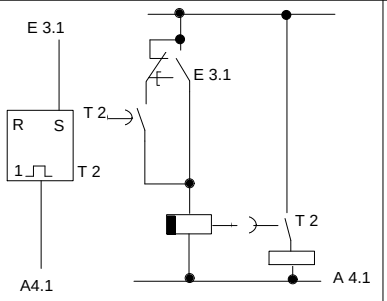


DU und DE sind digitale Ausgänge der Zeitzelle. Am Ausgang DU steht der Zeitwert dualcodiert, am Ausgang DE BCD-codiert mit Zeitraster an.

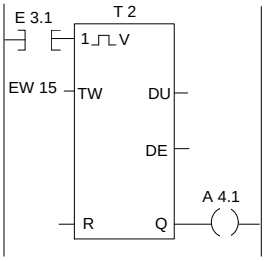
Zeitoperationen (Fortsetzung)

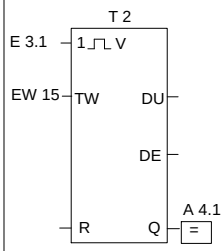
Verlängerter Impuls

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U     E   3.1
L     EW 15
SV    T   2
U     T   2
=     A   4.1
```

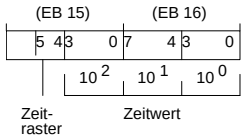




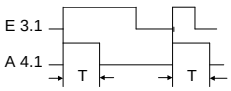
Bei Verknüpfungsergebnis "1" und erstmaliger Bearbeitung wird das Zeitglied gestartet.

Bei Verknüpfungsergebnis "0" bleibt das Zeitglied unbeeinflußt

Die Abfragen U T oder OT liefern Signalzustand "1", solange die Zeit läuft.



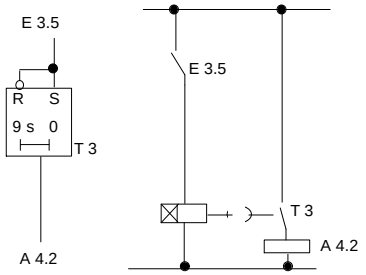
EW 15:
Setzen des Zeitwerts mit dem im BCD-Code vorliegenden Wert der Operanden E, A, M oder D (im Beispiel Eingangswort 15)



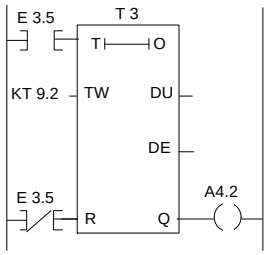
Zeitoperationen (Fortsetzung)

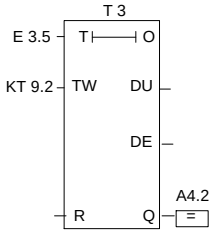
Einschaltverzögerung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U   E   3.5
L   KT  9.2
SE  T   3
UN  E   3.5
R   T   3
U   T   3
=   A   4.2
```





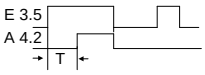
Bei Verknüpfungsergebnis "1" und erstmaliger Bearbeitung wird das Zeitglied gestartet. Bei wiederholter Bearbeitung mit Verknüpfungsergebnis "1" bleibt das Zeitglied unbeeinflußt.

Bei Signalzustand "0" am Eingang E 3.5 wird das Zeitglied auf Null gesetzt (gelöscht).

Die Abfragen U T bzw. O T liefern Signalzustand "1", wenn die Zeit abgelaufen ist und der Signalzustand "1" am Eingang E 3.5 noch ansteht.

KT 9.2:

Das Zeitglied wird mit dem angegebenen Wert (9) geladen. Die Zahl rechts vom Punkt gibt das Zeitraster an:

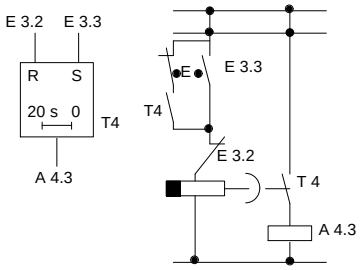


0 = 0.01 s 2 = 1 s
1 = 0.1 s 3 = 10 s

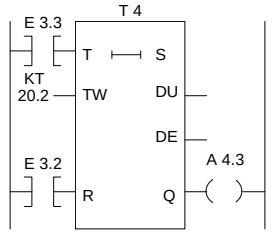
Zeitoperationen (Fortsetzung)

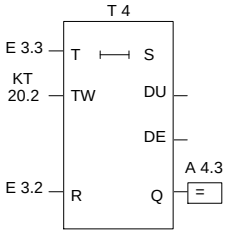
Speichernde Einschaltverzögerung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U   E   3.3
L   KT  20.2
SS  T   4
U   E   3.2
R   T   4
U   T   4
=   A   4.3
```

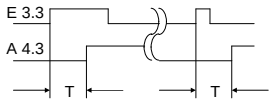




Bei Verknüpfungsergebnis "1" und erstmaliger Bearbeitung wird das Zeitglied gestartet.

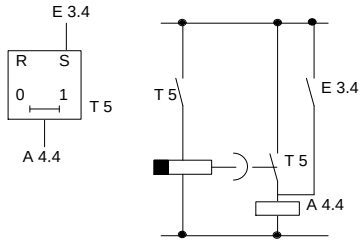
Bei Verknüpfungsergebnis "0" bleibt das Zeitglied unbeeinflußt.

Die Abfragen UT bzw. OT liefern Signalzustand "1", wenn die Zeit abgelaufen ist. Der Signalzustand wird erst dann "0",wenn das Zeitglied mit der Funktion RT zurückgesetzt worden ist.

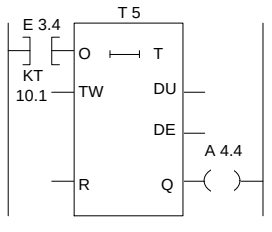


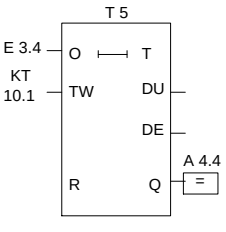
Ausschaltverzögerung

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan




```
U   E   3.4
L   KT  10.1
SA  T   5
U   T   5
=   A   4.4
```

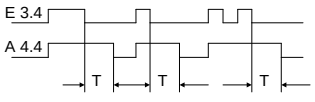




Wenn das Verknüpfungsergebnis am Starteingang von "1" nach "0" wechselt, wird die Zeit gestartet. Sie läuft mit der programmierten Zeitdauer ab.

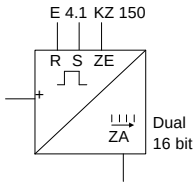
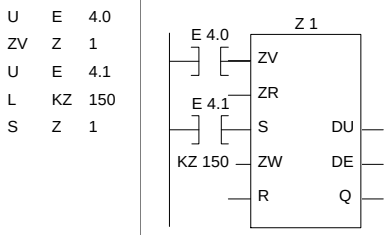
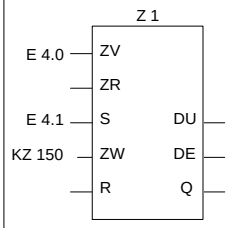
Bei Verknüpfungsergebnis "1" wird das Zeitglied auf Null gesetzt (gelöscht).

Die Abfragen UT bzw. OT liefern Signalzustand "1", wenn die Zeit läuft oder das Verknüpfungsergebnis am Eingang "1" ist.



Zähloperationen

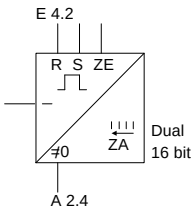
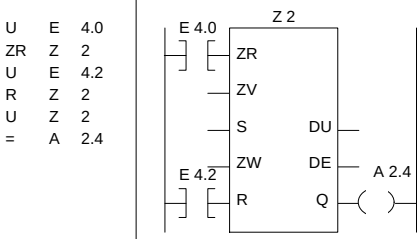
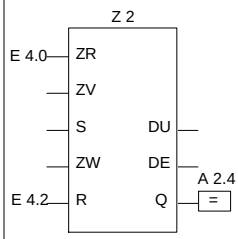
Zähler setzen

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungsliste	Kontaktplan	Funktionsplan
			

Wenn das Verknüpfungsergebnis am Starteingang (E 4.1) von "0" nach "1" wechselt, wird der Zähler mit dem angegebenen Wert (150) geladen.

Der für die Flankenauswertung des Setzeingangs erforderliche Merker ist im Zählwort mitgeführt.
DU und DE sind digitale Ausgänge der Zählerzelle. Am Ausgang DU steht der Zählwert dualcodiert, am Ausgang DE BCD-codiert an.

Zähler rücksetzen

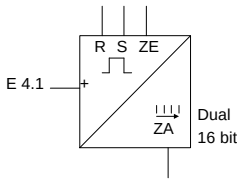
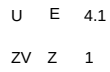
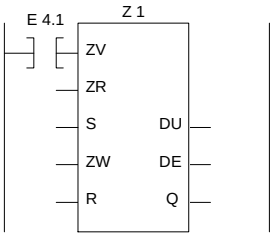
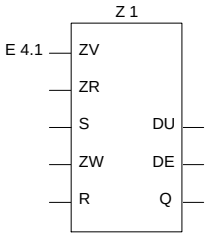
Aufgabenstellung	STEP-5-Darstellung		
	Anweisungsliste	Kontaktplan	Funktionsplan
			

Bei Verknüpfungsergebnis "1" (E 4.2) wird der Zähler auf Null gesetzt (rückgesetzt).

Bei Verknüpfungsergebnis "0" bleibt der Zähler unbeeinflußt.

Zähloperationen (Fortsetzung)

Vorwärts zählen

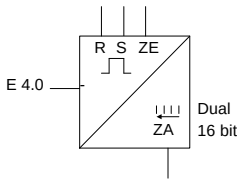
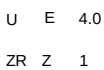
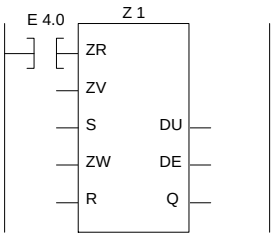
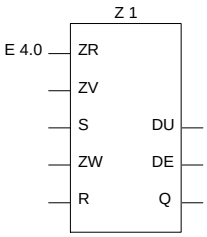
Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
			

Der Wert des adressierten Zählers wird um 1 erhöht, maximal bis zum Zählwert 999. Die Funktion ZV wird nur bei einer positiven Flanke (von "0" nach "1") der vor ZV programmierten Verknüpfung ausgeführt. Die für die Flankenauswertung der Zähleingänge erforderlichen Merker sind im Zählwort mitgeführt.

Durch die zwei getrennten Flankenmerker für ZV und ZR kann ein Zähler mit zwei verschiedenen Eingängen als Vorwärts-/ Rückwärtszähler verwendet werden.

Zähloperationen (Fortsetzung)

Rückwärts zählen

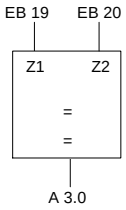
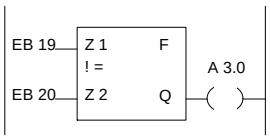
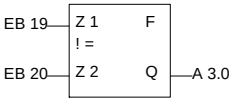
Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
			

Der Wert des adressierten Zählers wird um 1 erniedrigt, maximal bis zum Zählwert 0. Die Funktion wird nur bei einer positiven Flanke (von "0" nach "1") der vor ZR programmierten Verknüpfung wirksam. Die für die Flankenauswertung der Zähleingänge erforderlichen Merker sind im Zählwort mitgeführt.

Durch die zwei getrennten Flankenmerker für ZV und ZR kann ein Zähler mit zwei verschiedenen Eingängen als Vorwärts-/Rückwärtszähler verwendet werden.

Vergleichsoperationen

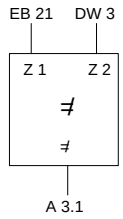
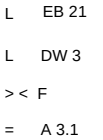
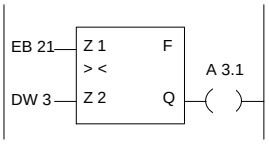
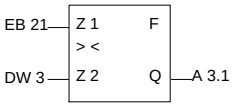
Vergleich auf gleich

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
	L EB 19 L EB 20 ! = F = A 3.0		

Der zuerst angegebene Operand wird mit dem nachfolgenden Operanden entsprechend der Vergleichsfunktion verglichen.
Der Vergleich ergibt ein binäres Verknüpfungsergebnis.
VKE = "1": Vergleich ist erfüllt, wenn AKKU-1-L gleich AKKU-2-L
VKE = "0": Vergleich ist nicht erfüllt, wenn AKKU-1-L ungleich AKKU-2-L
Die Anzeigen ANZ 1 und ANZ 0 werden, wie in der Operationsliste beschrieben, gesetzt.
AKKU-2-H und AKKU-1-H bleiben beim 16-bit-Festpunktvergleich an der Operation unbeteiligt.
Beim 32-bit-Festpunktvergleich (!=D) und Gleitpunktvergleich (!=G) werden die gesamten Inhalte von AKKU 1 und AKKU 2 (32 bit) miteinander verglichen.
Beim Vergleich wird die Zahlendarstellung der Operanden berücksichtigt, d. h., der Inhalt von AKKU-1-L und AKKU-2-L wird hier als Festpunktzahl interpretiert.

Vergleichsoperationen
(Fortsetzung)

Vergleich auf ungleich

Aufgabenstellung	STEP-5-Darstellung		
	Anweisungs-liste	Kontaktplan	Funktionsplan
			

Der zuerst angegebene Operand wird mit dem nachfolgenden Operanden entsprechend der Vergleichsfunktion verglichen.
Der Vergleich ergibt ein binäres Verknüpfungsergebnis.
VKE = "1": Vergleich ist erfüllt, wenn AKKU-1-L ungleich AKKU-2-L
VKE = "0": Vergleich ist nicht erfüllt, wenn AKKU-1-L gleich AKKU-2-L
Die Anzeigen ANZ 1 und ANZ 0 werden, wie am Anfang des Kapitels 3.5 beschrieben, gesetzt.
AKKU-2-H und AKKU-1-H bleiben beim 16-bit-Festpunktvergleich an der Operation unbeteiligt.
Beim 32-bit-Festpunktvergleich und Gleitpunktvergleich sind auch AKKU-2-H und AKKU-1-H beim Vergleich beteiligt.
Entsprechendes gilt für die Vergleiche auf größer, größer gleich, kleiner und kleiner gleich (siehe Operationsliste).
Beim Vergleich wird die Zahlendarstellung der Operanden berücksichtigt, d. h., der Inhalt von AKKU-1-L und AKKU-2-L wird hier als Festpunktzahl interpretiert.

3.5.3 Ergänzende Operationen

Den ergänzenden Operationsvorrat können Sie nur für Funktionsbausteine (FB und FX) verwenden. Der Gesamtoperationsvorrat für Funktionsbausteine besteht daher aus den Grundoperationen und den ergänzenden Operationen.

Zu den ergänzenden Funktionen gehören auch Systemoperationen: Mit den Systemoperationen können Sie z. B. den Speicher an beliebiger Stelle überschreiben oder den Inhalt der Arbeitsregister der CPU verändern.

Beachten Sie zum Thema "Systemoperationen" das Kapitel 9 "Speicherzugriffe".

Vorsicht
Systemoperationen sollten nur von erfahrenen Programmierern und Systemkennern und nur mit Vorsicht angewendet werden.

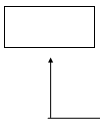
Bei den Funktionsbausteinen werden die Operationen nur in AWL dargestellt. Die Programme der Funktionsbausteine können also nicht in grafischer Form (KOP oder FUP) programmiert werden.
Im folgenden werden die ergänzenden Operationen beschrieben. Zusätzlich sind die Kombinationsmöglichkeiten der Substitutionsbefehle mit den Aktualoperanden angegeben.

Systemoperationen

Systemoperationen sind in der ersten Spalte der Tabellen mit **S** gekennzeichnet!

Binäre Verknüpfungen

Tabelle 3-11 Binäre Verknüpfungen mit Formaloperanden

Operation		Operand	Funktion
	U =		UND-Funktion, Abfrage eines Formaloperanden auf Signalzustand ’1’
	UN =		UND-Funktion, Abfrage eines Formaloperanden auf Signalzustand ’0’
	O =		ODER-Funktion, Abfrage eines Formaloperanden auf Signalzustand ’1’
	ON =		ODER-Funktion, Abfrage eines Formaloperanden auf Signalzustand ’0’ Formaloperand einsetzen Als Aktualoperanden sind binär adressierte Eingänge, Ausgänge, Daten und Merker (Parameterart: E, A ; Parametertyp: BI) sowie Zeiten und Zähler (Parametertyp: T, Z) zugelassen.

Digitalverknüpfungen

Tabelle 3-12 Digitalverknüpfungen

Operation		Operand	Funktion
	UW		UND-Verknüpfung von AKKU-1-L und AKKU-2-L
	OW		ODER-Verknüpfung von AKKU-1-L und AKKU-2-L
	XOW		Exklusiv-ODER-Verknüpfung von AKKU-1-L und AKKU-2-L

Die AKKUs 2, 3 und 4 werden nicht beeinflußt, jedoch die Anzeigen ANZ 1 und ANZ 0 (siehe Wort-Ergebnisanzeigen).

Speicheroperationen

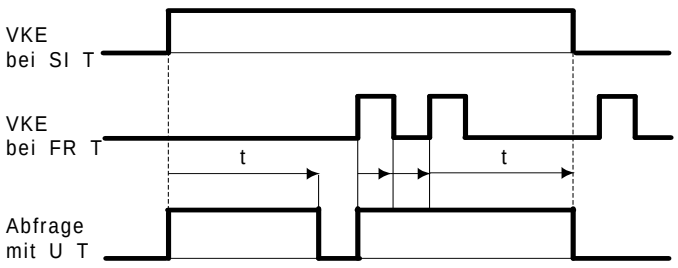
Tabelle 3-13 Speicheroperationen mit Formaloperanden

Operation	Operand	Funktion
S =		Setzen (binär) eines Formaloperanden
RB =		Rücksetzen (binär) eines Formaloperanden
RD=		Rücksetzen (digital) eines Formaloperanden für Zeiten und Zähler
= =		Zuweisen des Verknüpfungsergebnisses an einen Formaloperanden
		Formaloperand einsetzen
		Als Aktualoperanden sind binär adressierte Eingänge, Ausgänge und M-Merker zugelassen (Parameterart: E, A; Parametertyp: BI).

Zeit- und Zähloperationen

Tabelle 3-14 Zeit- und Zähloperationen mit Formaloperanden

Operation		Operand	Funktion
	SI =		Eine Zeit (Formaloperand) als Impuls starten. Der Zeitwert muß in AKKU-1-L hinterlegt sein (Parameterart T) .
	SE =		Eine Zeit (Formaloperand) einschaltverzögernd starten. Der Zeitwert muß in AKKU-1-L hinterlegt sein (Parameterart T).
	SVZ =		Eine Zeit (Formaloperand) als verlängerten Impuls starten mit dem im AKKU-1-L hinterlegten Zeitwert/einen Zähler (Formaloperand) setzen mit dem im AKKU-1-L hinterlegten Zählwert; (Parameterart: T, Z).
	SSV =		Eine Zeit (Formaloperand) als speichernde Einschaltverzögerung starten mit dem im AKKU-1-L hinterlegten Wert bzw. Vorwärtszählen eines Zählers (Formaloperand); (Parameterart: T, Z).
	SAR =		Eine Zeit (Formaloperand) als Ausschaltverzögerung starten mit dem im AKKU-1-L hinterlegten Wert bzw. Rückwärtszählen eines Zählers (Formaloperand); (Parameterart: T, Z).
	FR =		Formaloperanden (Zeit/Zähler) für den Neustart freigeben (siehe "FR T", "FR Z"); (Parameterart: T, Z). Formaloperand einsetzen
	FR	T 0 bis 255	Eine Zeit für den Neustart freigeben: Die Operation wird **nur** bei steigender Flanke des VKE ausgeführt (Wechsel von ’0’ auf ’1’). Die Zeit wird neu gestartet, wenn bei der Startoperation das VKE = ’1’ ist. (siehe Skizze mit Zeitdiagramm unter der Tabelle)
		Z 0 bis 255	Einen Zähler für das Setzen bzw. Zählen freigeben: Die Operation wird **nur** bei steigender Flanke des VKE ausgeführt (Wechsel von ’0’ auf ’1’). Der Zähler wird neu bearbeitet, wenn bei der Zähloperation das VKE = ’1’ ist.



Beispiele

a)

Funktionsbausteinaufruf	Programm im Funktionsbaustein	ausgeführtes Programm
:SPA FB 203		
NAME :BEISP1		
ANNA : E 10.3	:U =ANNA	:U E 10.3
BERT : T 17	:L KT 010.2	:L KT 010.2
HANS : A 18.4	:SSV =BERT	:SS T 17
	:U =BERT	:U T 17
	:= =HANS	:= A 18.4

b)

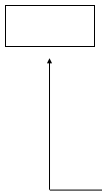
Funktionsbausteinaufruf	Programm im Funktionsbaustein	ausgeführtes Programm
:SPA FB 204		
NAME :BEISP2		
MAXI : E 10.5	:U =MAXI	:U E 10.5
IRMA : E 10.6	:SSV =DORA	:ZV Z 15
EVA : E 10.7	:U =IRMA	:U E 10.6
DORA : Z 15	:SAR =DORA	:ZR Z 15
EMMA : M 58.3	:U =EVA	:U E 10.7
	:L KZ 100	:L KZ 100
	:SVZ =DORA	:S Z 15
	:UN =DORA	:UN Z 15
	:= =EMMA	:= M 58.3

c)

Funktionsbausteinaufruf	Programm im Funktionsbaustein	ausgeführtes Programm
:SPA FB 205		
NAME :BEISP3		
KURT : E 10.4	:U =KURT	:U E 10.4
KARL : T 18	:L =EGON	:L EW 20
EGON : EW 20	:SVZ =KARL	:SV T 18
MAUS : M 100.7	:U =KARL	:U T 18
	:= =MAUS	:= M 100.7

Lade- und Transferoperationen

Tabelle 3-15 Lade- und Transferoperationen mit Formaloperanden

Operation	Operand	Beschreibung
L =		Laden eines Formaloperanden: Der Wert des als Formaloperand vorgegebenen Operanden wird in den AKKU geladen (Parameterart: E, T, Z, A; Parametertyp: BY, W, D).
LC =		Codiertes Laden eines Formaloperanden: Der Wert der als Formaloperand vorgegebenen Zeit- oder Zählzelle wird BCD-codiert in den AKKU geladen (Parametertyp: T, Z).
LW =		Laden des Bitmusters eines Formaloperanden: Das Bitmuster des Formaloperanden wird in den AKKU geladen (Parameterart: D; Parametertyp: KF, KH, KM, KY, KC, KT, KZ).
LD =		Laden des Bitmusters eines Formaloperanden: Das Bitmuster des Formaloperanden wird in den AKKU geladen (Parameterart: D; Parametertyp: KG).
T =		Transferieren zu einem Formaloperanden: Der Akkumulatorinhalt wird zu dem als Formaloperand vorgegebenen Operanden transferiert (Parameterart: E, A; Parametertyp: BY, W; D). Formaloperand einsetzen

Als Aktualoperanden sind die den Grundoperationen entsprechenden Operanden – **ausgenommen S-Merker** – zugelassen. Bei ’LW=’ sind ein Datum in Form eines Binär- (KM) oder eines Hexadezimalmusters (KH), 2 byteweise Betragszahlen (KY), Zeichen (KC), Festpunktzahl (KF), Zeitwert (KT) und Zählwert (KZ) zugelassen. Bei ’LD=’ ist eine Gleitpunktzahl als Datum zugelassen.

Tabelle 3-16 Lade- und Transferoperationen mit speziellen Operanden

Operation		Operand		Funktion
	L	BA	0 bis 255	Laden eines Wortes in den AKKU 1 aus dem Bereich "Anschaltung" (BA-Bereich)
		BB	0 bis 255	Laden eines Wortes in den AKKU 1 aus dem erweiterten Bereich "Anschaltung" (BB-Bereich)
	L	BS	0 bis 255	Laden eines Wortes in den AKKU 1 aus dem Bereich "Systemdaten" (BS-Bereich)
		BT	0 bis 255	Laden eines Wortes in den AKKU 1 aus dem erweiterten Bereich "Systemdaten" (BT-Bereich)
	T	BA	0 bis 255	Transferieren des Inhalts von AKKU 1 zu einem Wort des Bereichs "Anschaltung" (BA-Bereich)
		BB	0 bis 255	Transferieren des Inhalts von AKKU 1 zu einem Wort des erweiterten Bereichs "Anschaltung" (BB-Bereich)
	T	BS	60 bis 63	Transferieren des Inhalts von AKKU 1 zu einem Wort des Bereichs "Systemdaten" (BS-Bereich)
		BT	0 bis 255	Transferieren des Inhalts von AKKU 1 zu einem Wort des erweiterten Bereichs "Systemdaten" (BT-Bereich)

Im Gegensatz zu den Bereichen BA, BB und BT dürfen vom BS-Bereich nur die Wörter BS 60 bis BS 63 für Anwenderzwecke frei genutzt werden. Beachten Sie dazu den Abschnitt 8.3.4 "BS-/BT-Bereich".

Den BT-Bereich können Sie in seiner gesamten Länge (BT 0 bis BT 255) benutzen, **vorausgesetzt**, Sie verwendenkeine Standard-Funktionsbausteine.

Rechenoperationen

Tabelle 3-17 Rechenoperation ENT

Operation		Operand	Funktion
	ENT	–	Es findet ein Stack-Lift in die AKKUs 3 und 4 statt: <AKKU 4> := <AKKU 3> <AKKU 3> := <AKKU 2> <AKKU 2> := <AKKU 2> <AKKU 1> := <AKKU 1> Die AKKUs 1 und 2 werden nicht verändert. Der alte Inhalt des AKKUs 4 geht verloren.

Beispiel

```
Folgender Bruch soll ausgerechnet werden: (30 + 3 * 4) / 6 = 7
```

	AKKU 1	AKKU 2	AKKU 3	AKKU 4
Vorbelegung der Akkus vor der aritmetischen Operationskette	a	b	c	d
L KF +30	30	a	c	d
L KF +3	3	30	c	d
ENT	3	30	30	c
L KF +4	4	3	30	c
x F	12	30	c	c
+ F	42	c	c	c
L KF +6	6	42	c	c
: F	7	c	c	c

Tabelle 3-18 Ergänzende arithmetische Operationen

Operation		Operand		Funktion
S	ADD	BF	-128 bis +127	Addieren einer Byte-Konstanten (Festpunkt) zum AKKU-1-L (vorzeichenexpandiert); die Anzeigen in ANZ 0, ANZ 1, OV und OS werden nicht beeinflußt. – AKKU-1-H sowie die AKKUs 2 bis 4 bleiben unverändert.
S	ADD	KF	-32 768 bis +32 767	Addieren einer Festpunktkonstanten (Wort) zum AKKU-1-L; die Anzeigen in ANZ 0, ANZ 1, OV und OS werden nicht beeinflußt. – AKKU-1-H sowie die AKKUs 2 bis 4 bleiben unverändert.
S	ADD [1]	DH	0000 0000 bis FFFF FFFF	Addieren einer Doppelwort-Festpunktkonstanten zum AKKU 1; die Anzeigen in ANZ 0, ANZ 1, OV und OS werden nicht beeinflußt. – Die AKKUs 2 bis 4 bleiben unverändert.
S	+D [1]			Addieren zweier Doppelwort-Festpunktzahlen (AKKU 2 + AKKU 1); dasErgebnis ist über ANZ 0/ANZ 1 auswertbar. [2]
S	-D [1]			Subtrahieren zweier Doppelwort-Festpunktzahlen (AKKU 2 - AKKU 1); das Ergebnis ist über ANZ 0/ANZ 1 auswertbar. [2]
S	TAK			Tauschen der Inhalte von AKKU 1 und AKKU 2

[1] Die Programmierung ist abhängig vom PG-Typ und vom Ausgabestand der PG-Systemsoftware.

[2] Veränderungen von AKKU 2 und AKKU 3: siehe Abschnitt 3.5.1 "Grundoperationen/arithmetische Operationen".

3.5.4 Organisatorische Operationen

Zu den organisatorischen Operationen gehören auch Systemoperationen.

Vorsicht
Systemoperationen sollten nur von erfahrenen Programmierern und Systemkennern und nur mit Vorsicht angewendet werden.

Systemoperationen sind in der ersten Spalte der Tabellen mit S gekennzeichnet!

Sprungoperationen

Das Sprungziel für unbedingte und bedingte Sprünge wird symbolisch angegeben (maximal 4 Zeichen, beginnend mit einem Buchstaben). Dabei ist der Symbolparameter des Sprungbefehls identisch mit der Symboladresse der anzuspringenden Anweisung. Bei der Programmierung muß berücksichtigt werden, daß die absolute Sprungdistanz nicht mehr als ± 127 Wörter umfaßt und eine STEP-5-Anweisung aus mehr als einem Wort bestehen kann. Sprünge dürfen nur innerhalb eines Bausteins durchgeführt werden; Sprünge über "Netzwerke" hinweg sind unzulässig ("Netzwerk" = Gliederungselement bei PB, SB und OB, siehe PG-Beschreibung).

Hinweis
Sprunganweisung und Sprungziel (Symboladresse) müssen im **gleichen** Netzwerk liegen. Pro Netzwerk darf der Name einer Symboladresse nur **einmal** vergeben werden.
Ausnahme: Dies gilt nicht für den Sprung SPR, bei dem als Parameter eine absolute Sprungdistanz angegeben wird.

Tabelle 3-19 Sprungoperationen

Operation	Operand	Funktion
SPA =	adr (adr =Symbol-adresse	Sprung unbedingt: Der unbedingte Sprung wird unabhängig von Bedingungen ausgeführt.
SPB =	mit maximal 4 Zeichen)	Sprung bedingt: Der bedingte Sprung wird ausgeführt, wenn VKE = 1 ist. Bei VKE = 0 wird die Anweisung nicht ausgeführt und das Verknüpfungsergebnis auf VKE = 1 gesetzt.
SPZ =		Sprung bei Ergebnis ’0’ : Der Sprung wird nur dann ausgeführt, wenn ANZ 1 = 0 und ANZ 0 = 0 ist. Das Verknüpfungsergebnis wird nicht verändert.

	Operation	Operand	Funktion
Fortsetzung der Tabelle 3-19:			
	SPN =	adr (adr = Symbol-adresse mit maximal 4 Zeichen)	Sprung bei Ergebnis ≠ 0 : Der Sprung wird nur dann ausgeführt, wenn ANZ 1 ≠ ANZ 0 ist. Das Verknüpfungsergebnis wird nicht verändert.
	SPP =		Sprung bei Ergebnis > ’0’ : Der Sprung wird nur dann ausgeführt, wenn ANZ 1 = 1 und ANZ 0 = 0 ist. Das Verknüpfungsergebnis wird nicht verändert.
	SPM =		Sprung bei Ergebnis < ’0’: Der Sprung wird nur dann ausgeführt, wenn ANZ 1 = 0 und ANZ 0 = 1 ist. Das Verknüpfungsergebnis wird nicht verändert.
	SPO =		Sprung bei Überlauf (Overflow): Der Sprung wird ausgeführt, wenn die Anzeige OV = 1 ist. Wenn kein Überlauf vorliegt (OV = 0), wird der Sprung nicht ausgeführt. Das Verknüpfungsergebnis wird nicht verändert. Ein Überlauf entsteht, wenn bei gegebener Zahlendarstellung der zulässige Bereich durch eine arithmetische Operation überschritten wird.
	SPS =		Sprung, wenn die Anzeige OS (Overflow speichernd) gesetzt ist : Der Sprung wird ausgeführt, wenn die Anzeige OS = 1 ist. Wenn kein Überlauf vorliegt (OS = 0), wird der Sprung nicht ausgeführt. Das Verknüpfungsergebnis wird nicht verändert. Ein Überlauf entsteht, wenn bei gegebener Zahlendarstellung der zulässige Bereich im Verlauf mehrerer arithmetischer Operationen überschritten wird.
S	SPR	-32 768 bis +32 767	Beliebiger relativer Sprung innerhalb des Anwenderspeichers bzw. innerhalb eines Funktionsbausteins (z. B. um in ein anderes Netzwerk zu gelangen). Die Operation wird immer ausgeführt, unabhängig von Bedingungen. Als Operand muß die Adressen-Differenz "Sprungziel - aktuelle Operation" (Anzahl Wörter) angegeben werden. Dabei wird der Sprung zu einer höheren (positiver Operand) oder niedrigeren (negativer Operand) Adresse als die der aktuellen Operation ausgeführt.

Vorsicht
Bei unsachgemäßer Anwendung derOperation **SPR** können undefinierte Zustände der Anlage auftreten! Sie sollte daher nur von sehr erfahrenen Programmierern und Systemkennern benutzt werden.

Schiebeoperationen

Tabelle 3-20 Schiebeoperationen

Operation	Operand	Funktion (Operation mit AKKU 1)
SLW	0 bis 15	Schieben nach links (von rechts werden Nullen nachgezogen)
SRW	0 bis 15	Schieben nach rechts (von links werden Nullen nachgezogen)
SLD	0 bis 32	Schieben eines Doppelwortes nach links (von rechts werden Nullen nachgezogen)
SVW	0 bis 15	Schieben mit Vorzeichen nach rechts (von links wird Bit 15 nachgezogen)
SVD	0 bis 32	Schieben eines Doppelwortes mit Vorzeichen nach rechts (von links wird Bit 31 nachgezogen)
RLD	0 bis 32	Rotieren nach links
RRD	0 bis 32	Rotieren nach rechts

Bei den Schiebeoperationen ist nur der AKKU 1 an der Ausführung beteiligt. Der Parameterteil dieser Operationen gibt an, um wieviele Stellen der AKKU-Inhalt geschoben bzw. rotiert wird. Bei SLW, SRW und SVW ist nur das niederwertige Wort an den Schiebeoperationen beteiligt, bei SLD, SVD, RLD und RRD der gesamte Inhalt des AKKU 1 (32 bit).

Die Schiebeoperationen werden unabhängig von Bedingungen ausgeführt.

Der Wert des zuletzt hinausgeschobene Bits kann über ANZ 1/ANZ 0 mit Sprungoperationen abgefragt werden:

Schieben: letztes geschobenes Bit	ANZ 1	ANZ 0	Sprungoperation
0	0	0	SPZ=
1	1	0	SPN= SPP=

Beispiele

1. Der Inhalt des Datenwortes DW 52 soll gelesen, um 4 bit nach links verschoben und dann im Datenwort DW 53 abgelegt werden.

```
   STEP-5-Programm:   Inhalt der Datenwörter:

   :L    DW 52        KH = 14AF
   :SLW  4
   :T    DW 53        KH = 4AF0
```

2. Das Eingangs-Doppelwort ED 0 soll gelesen, und der Inhalt von AKKU 1 so verschoben werden, daß die fettgedruckten Bitstellen des Eingangs-Doppelwortes (z. T. verschoben) erhalten bleiben und die übrigen Bitstellen mit definierten Werten (0 H bzw. 0F H) besetzt werden.

```
   STEP-5-Programm:   Inhalt von AKKU 1 (hexadezimal)

                      AKKU-1-H:            AKKU-1-L:

   :L    ED 0         2348                 ABCD
   :SLW     4         2348                 BCD0
   :SRW     4         2348                 0BCD
   :SLD     4         3480                 BCD0
   :SVW     4         3480                 FBCD
   :SVD     4         0348                 0FBC
   :RLD     4         3480                 FBC0
   :RRD     4         0348                 0FBC
```

3. Multiplikation mit 2er-Potenz, z. B. neuer Wert = alter Wert x 8

```
   :L    MW 10
   :SLW     3
   :T    MW 10        Achtung: Positive Bereichsgrenze
                               nicht überschreiten!
```

4. Division durch 2er-Potenz, z. B. neuer Wert = alter Wert : 4

```
   :A    DB 5
   :L    DW 0
   :SRW     2
   :T    DW 0
```

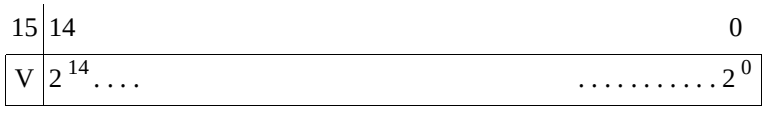
Umwandlungsoperationen

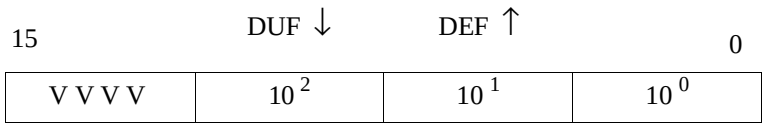
Tabelle 3-21 Umwandlungsoperationen

Operation		Funktion
	KEW	Bildung des 1er-Komplements von AKKU-1-L (16 bit)
	KZW	Bildung des 2er-Komplements von AKKU-1-L (16 bit)
	KZD	Bildung des 2er-Komplements von AKKU 1 (32 bit)
	DEF	Festpunktwandlung (16 bit) von BCD in dual
	DUF	Festpunktwandlung (16 bit) von dual in BCD
	DED	Doppelwortwandlung (32 bit) von BCD in dual
	DUD	Doppelwortwandlung (32 bit) von dual in BCD
	FDG	Wandlung einer Festpunktzahl (32 bit) in eine Gleitpunktzahl (32 bit); siehe OB 220 : Vorzeichenerweiterung
	GFD	Wandlung einer Gleitpunktzahl in eine Festpunktzahl (32 bit)

DEF Der im AKKU-1-L (Bit 0 bis Bit 15) stehende Wert wird als BCD-codierte Zahl interpretiert. Nach der Umwandlung steht im AKKU-1-L eine 16-bit-Festpunktzahl.

DUF Der im AKKU-1-L (Bit 0 bis Bit 15) stehende Wert wird als 16-bit-Festpunktzahl interpretiert. Nach der Umwandlung steht im AKKU-1-L eine BCD-codierte Zahl.

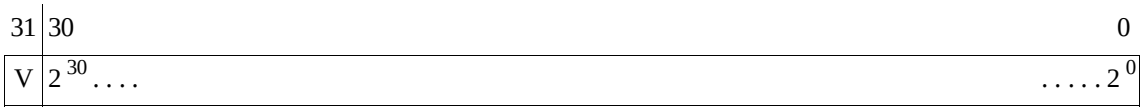




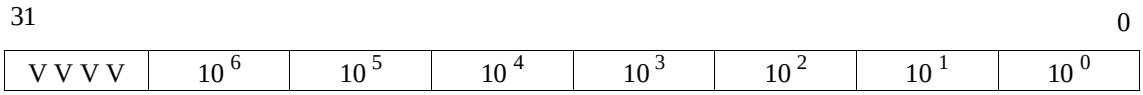
V (Vorzeichen): 0 = positiv
1 = negativ

DED Der im AKKU 1 (Bit 0 bis Bit 31) stehende Wert wird als BCD-codierte Zahl interpretiert. Nach der Umwandlung steht im AKKU 1 eine 32-bit-Festpunktzahl.

DUD Der im AKKU 1 (Bit 0 bis Bit 31) stehende Wert wird als 32-bit-Festpunktzahl interpretiert. Nach der Umwandlung steht im AKKU 1 eine BCD-codierte Zahl.



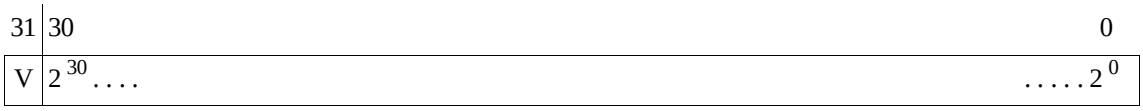
DUD ↓ DED ↑



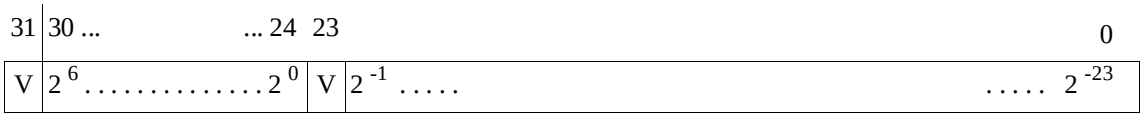
V (Vorzeichen): 0 = positiv
1 = negativ

FDG Der im AKKU 1 (Bit 0 bis Bit 31) stehende Wert wird als 32-bit-Festpunktzahl interpretiert. Nach der Umwandlung steht im AKKU 1 eine Gleitpunktzahl (Exponent und Mantisse).

GFD Der im AKKU 1 (Bit 0 bis Bit 31) stehende Wert wird als Gleitpunktzahl interpretiert. Nach der Umwandlung steht im AKKU 1 eine 32-bit-Festpunktzahl.



FDG ↓ GFD ↑



Exponent Mantisse

Die Umwandlung geschieht durch Multiplikation der (dualen) Mantisse mit dem Wert des (dualen) Exponenten, indem der Mantissenwert über einen gedachten Dezimalpunkt um den Wert des Exponenten (zur Basis ’2’) zu höherwertigen Bitstellen verschoben wird. Nach der Multiplikation bleiben Reste der ursprünglichen Mantisse rechts vom gedachten Dezimalpunkt übrig. Diese Bitstellen werden vom ganzzahligen Ergebnis abgeschnitten.

Durch diesen Umwandlungsalgorithmus ergeben sich folgende Ergebnisklassen:

- Gleitpunktzahlen ≥ **0 oder** ≤ **-1** ergeben die **nächst kleinere ganze Zahl**.
- Gleitpunktzahlen **< 0 und > -1** ergeben den **Wert '0'**.

Umwandlungsbeispiele

Gleitpunktzahl	**GFD**	32-bit-Festpunktzahl
+5,7	→	5
-2,3	→	-3
-0,6	→	**0**
+0,9	→	**0**

Beispiele zu KEW, KZW

```
1. Der Inhalt des Datenwortes 64 soll Bit für Bit
   invertiert ("umgekehrt") und in Datenwort 78
   abgelegt werden.

   STEP-5-Programm:   Belegung der Datenwörter:

   :L DW 64           KM = 0011111001011011
   :KEW
   :T DW 78           KM = 1100000110100100



2. Der Inhalt des Datenwortes 207 ist als Fest-
   punktzahl zu interpretieren und mit umgekehr-
   tem Vorzeichen im Datenwort 51 abzulegen.

   STEP-5-Programm:   Belegung der Datenwörter:

   :L DW 207          KF = +51
   :KZW
   :T DW 51           KF = -51
```

Dekrementieren/ Inkrementieren

Tabelle 3-22 Dekremtier-/Inkremtieroperation

Operation		Operand	Funktion
	D	1 bis 255	Dekrementieren des Low-Bytes (Bits 0 bis 7) von AKKU-1-L um den Operandenwert [1]
	I	1 bis 255	Inkrementieren des Low-Bytes (Bits 0 bis 7) von AKKU-1-L um den Operandenwert [1]

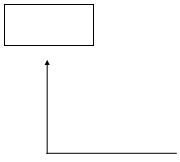
[1] Der Inhalt des Low-Bytes von AKKU-1-L wird um die als Operand angegebene Zahl ohne Übertrag dekrementiert (erniedrigt) bzw. inkrementiert (erhöht). Die Operationsausführung ist unabhängig von Bedingungen.

Beispiel

```
STEP-5-Programm:   Belegung der Datenwörter:

   :L DW 7         KH = 1010
   :I   16
   :T DW 8         KH = 1020
   :D   33
   :T DW 9         KH = 10FF
```

Bearbeitungsoperationen

Tabelle 3-23 Bearbeitungsoperationen

Operation		Operand	Funktion
	B	DW 0 bis 255	Bearbeite Datenwort: Die nachfolgend angegebene Operation wird mit dem im adressierten Datenwort angegebenen Parameter kombiniert und ausgeführt.
		MW 0 bis 254	Bearbeite Merkerwort: Die nachfolgend angegebene Operation wird mit dem im adressierten M-Merker angegebenen Parameter kombiniert und ausgeführt.
	B =		Bearbeite Fomaloperanden (Parameterart: B): Nur A DB, SPA PB, SPA OB, SPA FB, SPA SB können substituiert werden. Formaloperanden einsetzen

Operation		Operand	Funktion
Fortsetzung der Tabelle 3-23:			
S	BI	[1]	Indirekte Bearbeitung eines Formaloperanden: Eine Operation ausführen, deren Operationscode in einem Formaloperanden abgelegt ist. Die Nummer des auszuführenden Formaloperanden steht im AKKU 1.
	B	BS 60 bis 63 [1]	Eine Operation, die im Bereich Systemdaten (BS) steht, soll ausgeführt werden (freie Systemdaten: BS 60 bis 63). Bei 2-Wort-Operationen muß das 2. Wort nach BS n+1 geladen werden.

[1] Der Wert, der im Systemdatum oder im Formaloperanden steht, wird als Operationscode einer STEP-5-Operation interpretiert, die dann ausgeführt wird.

Hinweis
Mit **B DW**, **B MW**, **BI** oder **B BS** dürfen nur folgende Operationen kombiniert werden:

- U.. , UN.. , O.. , ON.. , S.. , R.. , =..
  mit den Bereichen E, A, M, S,
- FR T, R T, SA T, SE T, SI T, SS T, SV T,
- FR Z, R Z, S Z, ZR Z, ZV Z,
- L.., T.. mit den Bereichen P, Q, E, A, M, S, D, BA, BB, BS, BT,
- L T, L Z,
- LC T, LC Z,
- SPA=, SPB=, SPZ=, SPN=, SPP=, SPM=, SPO=,
- SLW, SRW,
- D, I, SES, SEF,
- A DB, SPA.. , SPB.., E DB, EX DX, AX DX, BAB FX, BA FX.

Das PG prüft die Zulässigkeit der Kombinationen nicht!

Beispiele zu den Bearbeitungs-Operationen

B DW/B MW

Operanden-Substitution

Mit den Anweisungen "B DW" und "B MW" können Sie, z. B. in einer Programmschleife, substituiert auf Daten zugreifen. Der substituierte Zugriff setzt sich zusammen aus der Anweisung B DW/B MW und einer unmittelbar nachfolgenden STEP-5-Operation aus dem o. g. Operationsspektrum.
"Substituiert" bedeutet, daß der Operand für die Operation nicht statisch beim Programmieren vorgegeben, sondern erst beim Ablauf Ihres STEP-5-Programms festgelegt wird.

Den Operanden**typ** wählen Sie beim Programmieren aus dem für die Operation zulässigen Spektrum aus, z. B. **PB** für die Operation "SPA PB nn".

Den Operanden**wert** (**nn** im Beispiel "SPA PB nn") müssen Sie vor einem substituierten Zugriff mit B DW/B MW in ein Daten- oder M-Merkerwort (Parameterwort) laden.

1. Prinzip der Substitution:

```
          :L    KF +120
          :T    MW 14      MW 14 mit dem Wert "KF  +120" laden
          :B    MW 14
          :L    EB 0
                   |
                   |_____________>  vor Ausfuehrung der Operation "L EB" wird der
                                    Operandenwert '0' durch den Wert '120' ersetzt;
                                    Ausfuehrung: L EB 120
```

2. Datenwort als Indexregister:

Es sollen die Inhalte der Datenwörter DW 20 bis DW 100 auf Signalzustand '0' gesetzt werden. Das Indexregister für den Parameter der Datenwörter ist DW 1.

```
          :L    KF +20     Versorgung des Indexregisters
          :T    DW 1
  M001    :L    KF +0      Rücksetzen
          :B    DW 1
          :T    DW 0
          :L    DW 1       Erhoehen des Indexregisters
          :L KF +1
          :+F
          :T    DW 1
          :L    KF +100
          :<=F
          :SPB  =M001      Sprung, wenn Index im Bereich liegt
          ...              weiteres STEP-5-Programm
```

Fortsetzung der Beispiele zur Operanden-Substitution:

3. Sprungverteiler für Unterprogrammtechnik:

```
           :B   MW 5
        ┌──:SPA  =M001      Inhalt Merkerwort MW 5:
+       │  :SPA  =M002
Sprung- │  :SPA  =M003            Sprungdistanz
distanz └─►:SPA  =M004           (maximal  ±127)
           :SPA  =M005
           : .
           : .
  M001     : .
           : .
           :BEA
  M002     : .            Vorteil:
           : .            Alle Teilprogramme stehen
           :BEA           in einem Baustein.
  M003     : .
           : .
           :BEA
```

4. Sprungverteiler für Bausteinaufrufe:

```
           :B   MW 10                    Inhalt Merkerwort MW 10:
           :SPA PB 0 ──►PB 0
                     ──►PB 1             Baustein-Nr. x
                     ──►PB 2
                     ──►PB 3
                     ──►.
                     ──►.
                     ──►PB x
```

Operanden-Substitution mit binären Operationen

Bei Operanden-Substitutionen mit **binären** Operationen können Sie folgende Operandentypen verwenden: Eingänge, Ausgänge, M-Merker, S-Merker, Zeiten und Zähler.
Der Aufbau des M-Merker- oder Datenwortes (Parameterwort) hängt bei dieser Substitution davon ab, welchen Operandentyp Sie verwenden.

Parameterwort für Ein und Ausgänge

Bit-Nr.	15 11	10 8	7	6 0
	ohne Bedeutung	Bit-Adresse von 0 bis 7	0	Byteadresse von 0 bis 127

Parameterwort für M-Merker

Bit-Nr.	15 11	10 8	7 0
	ohne Bedeutung	Bit-Adresse von 0 bis 7	Byteadresse von 0 bis 255

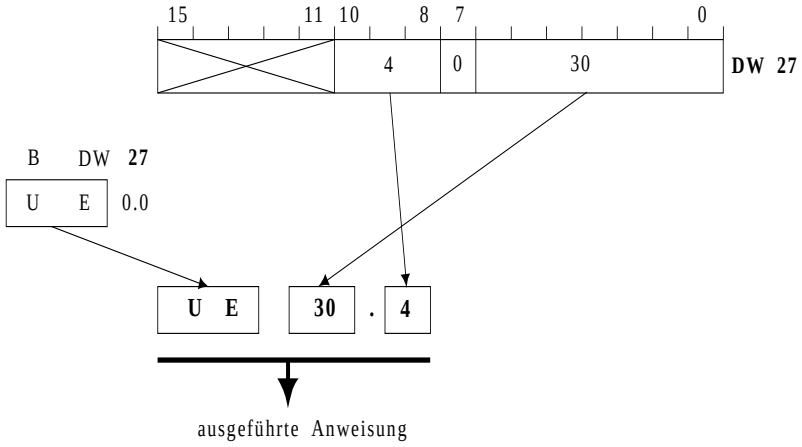
Parameterwort für S-Merker

Bit-Nr.	15	14 12	11 0
	0	Bit-Adresse von 0 bis 7	Byteadresse von 0 bis 4095

Parameterwort für Zeiten und Zähler

Bit-Nr.	15 8	7 0
	ohne Bedeutung	Nummer von Zeit- oder Zählerzelle von 0 bis 255

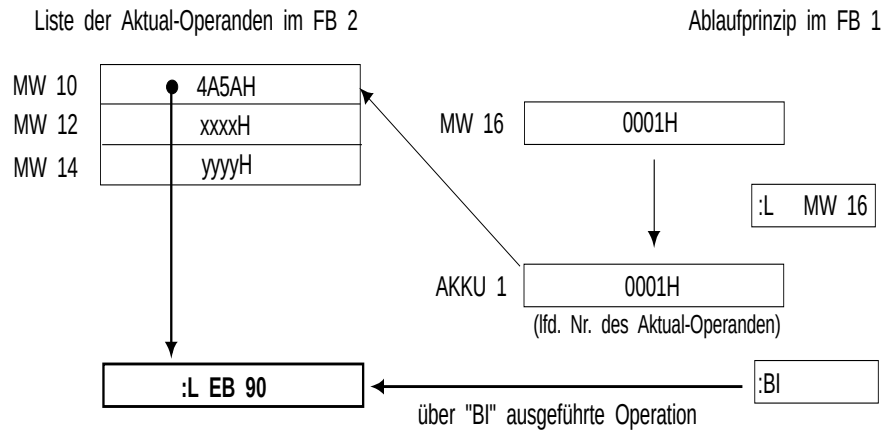
Prinzip der Substitution mit einer binären Operation



Beispiel zur BI-Operation

Im Funktionsbaustein FB 1 werden STEP-5-Operationen ausgeführt, deren Operations-Codes als Formaloperanden MW 10, MW 12 und MW 14 von einem aufrufenden Baustein übergeben werden.
Welcher der Operationscodes ausgeführt werden soll, wird vom aufrufenden Baustein als lfd. Nummer im Merkerwort **MW 16** hinterlegt.
Das Ergebnis der ausgeführten Operation befindet sich anschließend in AKKU 1 und wird in das Merkerwort MW 18 transferiert.

```
FB 1

NAME  :TEST

BEZ   :MW10      E/A/D/B/T/Z: D  KM/KH/KY/KC/KF/KT/KZ/KG:  KH
BEZ   :MW12      E/A/D/B/T/Z: D  KM/KH/KY/KC/KF/KT/KZ/KG:  KH
BEZ   :MW14      E/A/D/B/T/Z: D  KM/KH/KY/KC/KF/KT/KZ/KG:  KH

      :L    MW 16           lfd. Nr. des Formaloperanden mit dem
      :                     gewuenschten Operationscode
      :BI                   uebergebener Operationscode wird ausgefuehrt
      :T    MW 16           Ergebnis aus AKKU 1
      :BE

FB 2

      :
      :L    KF +1
      :T    MW 16           lfd. Nr. des Formaloperanden mit Operationscode
      :SPA  =AUFR
      :
      :
AUFR  :
      :SPA  FB 1            FB TEST aufrufen
NAME  :TEST
MW10  :     KH 4A5A         Operationscode "L EB 90", Formal-Operand 1
MW12  :     KH xxxx         anderer Operationscode,   Formal-Operand 2
MW14  :     KH yyyy         anderer Operationscode,   Formal-Operand 3
      :T    MW 18           AKKU 1 → MW 18
      :BE
```



Prozeßalarme sperren/freigeben

Tabelle 3-24 Prozeßalarme sperren/freigeben

Operation		Operand	Funktion
	AS		Prozeßalarmbearbeitung sperren
	AF		Prozeßalarmbearbeitung freigeben

"Prozeßlarme sperren/freigeben" kann angewendet werden, wenn die prozeßalarmgesteuerte Bearbeitung unterdrückt werden soll. In dem Programmteil, der zwischen den Anweisungen AS und AF steht, ist dann die prozeßalarmgesteuerte Bearbeitung nicht mehr möglich. Beachten Sie hierzu die Sonderfunktion OB 120 "Alarme gmeinsam sperren", Abschnitt 6.5.

3.5.5 Semaphor-Operationen

Benutzen zwei oder mehrere CPUs im Mehrprozessorbetrieb (siehe Kapitel 10) eines Automatisierungsgerätes bestimmte globale Speicherbereiche (Peripherie, CPs, IPs) gemeinsam, besteht die Gefahr, daß die CPUs einander Daten überschreiben oder daß ungültige Zwischenstände der Daten ausgelesen werden. Deshalb ist es erforderlich, den Zugriff der CPUs auf die gemeinsamen Speicherbereiche zu koordinieren.

Die Koordinierung der einzelnen CPUs ist mit Semaphoren durch die Operationen SES und SEF möglich:
Es kann z. B. folgende Koordinierung zweier CPUs programmiert werden: Nur nach erfolgreichem Setzen eines vereinbarten Semaphors (SES) greift jede der am Mehrprozessorbetrieb beteiligten CPUs auf den gemeinsamen Speicherbereich zu. Ein Semaphor xx kann dabei immer nur durch eine einzige CPU gesetzt werden. Gelingt einer CPU das Setzen des Semaphors nicht, so muß sie auf den Zugriff verzichten. Ebenso muß eine CPU auf einen weiteren Zugriff verzichten, nachdem sie den Semaphor wieder freigegeben hat (SEF).

SES/SEF Semaphor setzen/freigeben

(**keine** Systemoperationen)

Tabelle 3-25 Semaphor setzen/freigeben

Operation	Operand	Funktion
SES	0 bis 31	Setzen eines Semaphors
SEF	0 bis 31	Freigeben eines Semaphors Auswertung der Operationsergebnisse über ANZ 0/ANZ 1

Hinweis
Die Operationen SES xx und SEF xx müssen in **allen CPUs** programmiert werden, die synchronisiert auf einen **gemeinsamen** globalen Speicherbereich zugreifen sollen.

Standard-FB, Hantierungsbausteine und Bausteine für Mehrprozessor-Kommunikation verwalten die Koordinierung intern. Sie brauchen bei der Anwendung dieser Bausteine die Operationen SES xx und SEF xx nicht zu programmieren.

Wirkung von SES/SEF

Die Operation SES xx (Semaphor setzen) belegt für die befehlsausführende CPU ein bestimmtes Byte im Koordinator (**vorausgesetzt**, dieses ist nicht bereits durch eine andere CPU belegt). Solange sich die CPU dort eingetragen hat, dürfen die übrigen CPUs auf den mit dem Semaphor (Nummer 0 bis 31) geschützten Speicherbereich nicht mehr zugreifen. Der Bereich ist damit für alle anderen CPUs gesperrt.
Damit diese Koordinierung richtig funktioniert, müssen alle CPUs, die auf denselben Bereich auf dem globalen Speicher zugreifen wollen, denselben Semaphor benutzen.

Der Befehl SEF xx (Semaphor freigeben) setzt das Byte auf dem Koordinator wieder zurück. Dadurch wird der geschützte Speicherbereich für die anderen CPUs durch Semaphorsetzen wieder belegbar. Ein Semaphor kann nur von derjenigen CPU freigegeben werden, von der er gesetzt wurde.

Anwendung von SES/SEF

Bild 3-8 zeigt den prinzipiellen Ablauf einer Zugriffskoordinierung mit Hilfe eines Semaphors.

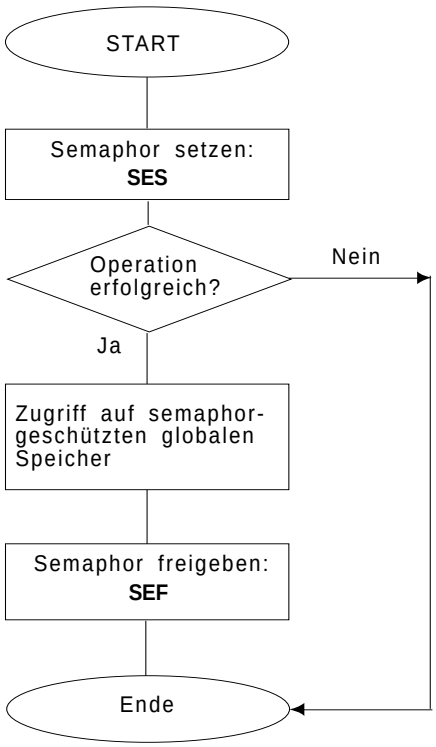


Bild 3-8 Koordinierung des Zugriffs auf den globalen Speicher

Vor jedem Setzen bzw. Freigeben eines bestimmten Semaphors fragen die Operationen SES und SEF den **Zustand** (= Status) dieses Semaphors ab. Die Anzeigen ANZ 0 und ANZ 1 werden dabei wie folgt beeinflußt:

ANZ 1	ANZ 0	Auswertung	Bedeutung
0	0	SPZ	Semaphor ist von anderer CPU gesetzt worden und kann nicht gesetzt/freigegeben werden
1	0	SPN, SPP	Semaphor wurde gesetzt/ freigegeben

Hinweis
Der Vorgang des Abfragens eines bestimmten Semaphors (= Lesevorgang) und der Vorgang des Setzens bzw. des Freigebens des Semaphors (= Schreibvorgang) bilden eine **Einheit**. Keine andere CPU kann während dieser Vorgänge auf diesen Semaphor zugreifen!

Zur Anwendung der Semaphoren beachten Sie bitte die folgenden Punkte:

- Ein Semaphor ist eine globale Größe, d. h. der Semaphor mit der Nummer 16 ist auch bei Einsatz von z. B. drei CPUs im gesamten System nur **einmal** vorhanden.

- Die Operationen SES und SEF müssen von **allen** CPUs verwendet werden, deren Zugriff auf einen gemeinsamen Speicherbereich koordiniert erfolgen soll.

- Alle beteiligten CPUs müssen immer die **gleiche** Anlaufart durchführen. Bei NEUSTART werden alle Semaphore gelöscht, bei einem manuellen oder automatischen Wiederanlauf bleiben sie erhalten.

- Der ANLAUF Mehrprozessorbetrieb muß synchronisiert erfolgen. Aus diesem Grund ist **kein** Testbetrieb erlaubt.

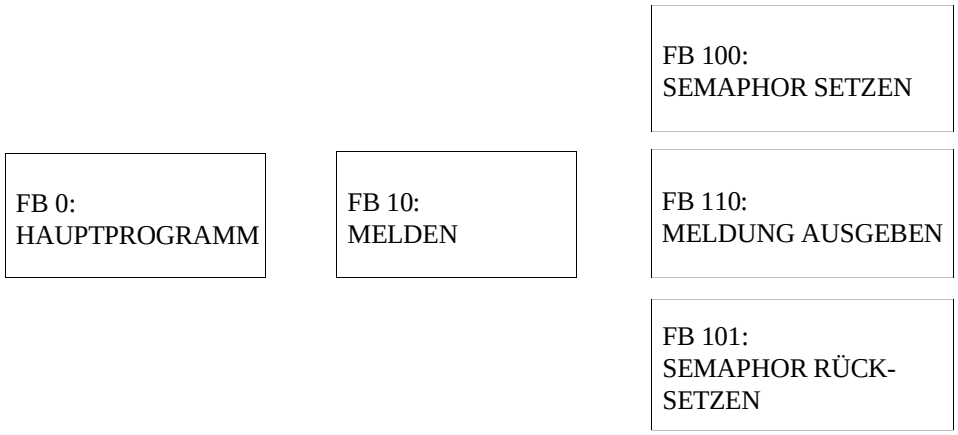
Anwendungsbeispiel für Semaphore

Aufgabenstellung:

In einem AG S5-135U stecken vier CPUs, die über einen gemeinsamen Speicherbereich der Q-Peripherie (QW 6) Statusmeldungen an ein Statusmeldegerät abgeben. Jede Statusmeldung muß 10 Sekunden lang ausgegeben werden und darf erst dann von einer neuen Meldung der gleichen oder einer anderen CPU überschrieben werden. Die Benutzung des Peripheriewortes QW 6 (erweiterte Peripherie, kein Prozeßabbild) wird über einen Semaphor gesteuert. Es darf nur diejenige CPU ihre Meldung auf das QW 6 schreiben, die durch erfolgreiches Setzen des zugeordneten Semaphors diesen Bereich für sich belegen konnte. Für die Dauer von jeweils 10 Sekunden bleibt der Semaphor gesetzt (TIMER T10). Erst nach Ablauf des Timers gibt die CPU den Semaphor und damit den belegten Bereich für die anderen CPUs wieder frei. Das QW 6 kann mit einer neuen Meldung beschrieben werden.
Ist beim Versuch einer CPU, den Semaphor zu setzen, dieser bereits durch eine andere CPU gesetzt worden, versucht die CPU im nächsten Zyklus erneut, den Semaphor zu setzen und ihre Meldung auszugeben.

Realisierung:

Das folgende Programm kann in allen vier CPUs – mit einer jeweils anderen Meldung – ablaufen. Folgende Bausteine werden geladen:



Es werden 5 Merker verwendet:

M 10.0 = 1: Eine Meldung ist angefordert oder in Bearbeitung.

M 10.1 = 1: Semaphor ist erfolgreich gesetzt worden.

M 10.2 = 1: Der Timer ist gestartet.

M 10.3 = 1: Die Meldung ist übertragen.

M 10.4 = 1: Semaphor ist rückgesetzt.

Fortsetzung auf der nächsten Seite

Fortsetzung des Semaphor-Anwendungsbeispiels:

FB 0

```
NAME  :MAIN

      :U    M 10.0
      :SPB  =M001         falls keine Meldung aktiv
      :
      :UN   E  0.0
      :BEB
      :
      :L    KH 2222       Meldung erzeugen und
      :T    MW 12
      :UN   M 10.0
      :S    M 10.0        Merker "MELDUNG" setzen.
      :
M001  :SPA  FB10          FB "MELDE" aufrufen
NAME  :MELDE
      :
      :BE
```

FB 10

```
NAME  :MELDE

      :UN   M 10.1        Wenn kein Semaphor gesetzt,
      :SPB  FB 100        FB "Semaphor setzen" aufrufen.
NAME  :SEMASET
      :
      :U    M 10.1        Wenn Semaphor gesetzt
      :UN   M 10.2        und Zeit nicht gestartet,
      :S    M 10.2
      :L    KT010.2       Zeit starten.
      :SV   T 10
      :
      :U    M 10.2        Wenn Zeit gestartet
      :UN   M 10.3        und keine Meldung uebertragen wird,
      :SPB  FB 110        FB "Meldung ausgeben" aufrufen.
NAME  :MELDAUSG
      :
      :U    M 10.2        Wenn Zeit gestartet
      :UN   M 10.4        und Semaphor nicht zurueckgesetzt
      :UN   T 10          und Zeit abgelaufen,
      :SPB  FB 101        FB "Semaphor ruecksetzen" aufrufen.
NAME  :SEMARESE
      :
      :UN   M  10.4       Wenn Semaphor rueckgesetzt,
      :BEB
      :
      :L    KH0000
      :T    MB10          alle Merker ruecksetzen.
      :BE
```

Fortsetzung auf der nächsten Seite

Fortsetzung des Semaphor-Anwendungsbeispiels:

FB100

```
NAME  :SEMASET

      :SES  10            Semaphor Nr. 10 setzen
      :SPZ  =M001
      :UN   M 10.1        Falls Semaphor erfolgreich gesetzt,
      :S    M 10.1        Merker "SEMAPHOR-GESETZT" setzen
M001  :BE
```

FB110

```
NAME: MELDAUSG

      :L    MW12          Meldung an die
      :T    QW 6          Peripherie uebertragen
      :UN   M 10.3
      :S    M 10.3        Merker "MELDUNG-UEBERTRAGEN"
      :                   setzen
      :BE
```

FB101

```
NAME  :SEMARESE

      :SEF  10            Semaphor Nr. 10 freigeben
      :SPZ  =M001
      :UN   M 10.4
      :S    M 10.4        Merker "SEMAPHOR-RUECKGESETZT"
      :                   setzen
M001  :BE
```

Betriebszustände und Programmbearbeitungsebenen

4

Inhalt von Kapitel 4

Betriebszustände und Programmbearbeitungsebenen

4

Dieses Kapitel gibt Ihnen eine Übersicht über die Betriebszustände und die Programmbearbeitungsebenen der CPU 928B. Es informiert Sie ausführlich über verschiedene Anlaufarten und damit verbundene Organisationsbausteine, in denen Sie Ihr spezifisches Programm für verschiedene Anlauffälle programmieren können.

Sie erfahren ferner, wodurch sich die Programmbearbeitungen "Zyklische Bearbeitung", "Zeitgesteuerte Bearbeitung" und "Alarmgesteuerte Bearbeitung" auszeichnen und welche Bausteine für Ihr Anwenderprogramm dabei zur Verfügung stehen.

4.1 Einführung und Übersicht

Die CPU 928B kennt drei Betriebszustände:

- Betriebszustand **STOP**
- Betriebszustand **ANLAUF**
- Betriebszustand **RUN**

In den Betriebszuständen ANLAUF und RUN können bestimmte Ereignisse auftreten, auf die das Systemprogramm reagieren muß. In vielen Fällen wird dabei ein für das Ereignis vorgesehener Organisationsbaustein (einer von OB 1 bis OB 35) als Anwenderschnittstelle aufgerufen.

Die Betriebszustände werden über LEDs auf der Frontplatte der CPU angezeigt.
Einige der Betriebszustände müssen Sie per Bedienung aktivieren. Dazu können Sie u. a. die Bedienungselemente auf der Frontplatte der CPU benutzen. Die Lage der LEDs und Bedienelemente können Sie Bild 4-1 entnehmen.

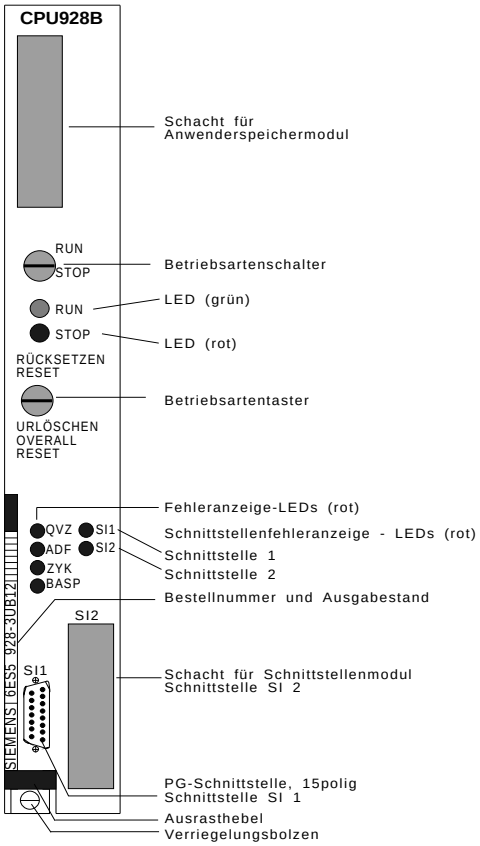


Bild 4-1 Frontplatte der CPU 928B mit Anzeige- und Bedienelementen

Anzeige der Betriebszustände durch Leuchtdioden (LEDs):

Auf der CPU-Frontplatte signalisieren Ihnen verschiedene LEDs den aktuellen Betriebszustand der CPU. Nachfolgende Tabelle zeigt den Zusammenhang zwischen den Anzeigen der STOP- und RUN-LED und dem zugehörigen Betriebszustand.
Diese Anzeigen werden ergänzt durch weitere LED-Anzeigen (BASP, ADF, QVZ, ZYK).

Tabelle 4-1 Bedeutung der LEDs "RUN" und "STOP"

LED "RUN"	LED "STOP"	Betriebszustand
an	aus	Die CPU ist im Betriebszustand RUN.
aus	an	Die CPU ist im Betriebszustand STOP. Nach einer Stoppanforderung per Schalter oder per PG-Funktion zeigt die STOP-LED Dauerlicht, da der Übergang in den Betriebszustand STOP hier vom Benutzer oder im Mehrprozessorbetrieb von einer anderen CPU angefordert und nicht von der CPU selbst verursacht wurde.
aus	aus	Die CPU ist im Betriebszustand ANLAUF, oder die CPU ist im ANLAUF/RUN, die Bearbeitungskontrolle ist aktiviert, und ein Haltepunkt ist erreicht, oder die CPU ist im ANLAUF/RUN, die Bearbeitungskontrolle ist aktiviert, und ein Haltepunkt wurde vor dem Erreichen wieder ausgetragen (Wartezustand).
aus	langsames Blinken	Die CPU ist im Betriebszustand STOP. Die CPU hat einen Übergang in die Betriebsart STOP (eventuell auch der anderen CPUs) verursacht. Ursachen sind z. B.: ADF, QVZ, LZF, BCF, Reglerfehler, Weckfehler, Zyklusfehler, BSTACK-Überlauf. Wenn Sie den Betriebsartenschalter auf STOP umlegen, geht das Blinken in Dauerlicht über.
aus	schnelles Blinken	Die CPU ist im Betriebszustand STOP. URLÖSCHEN wurde angefordert. Diese Anforderung kann von der CPU selbst oder aber per Bedienung erzeugt werden.
an	an	schwerer Systemfehler Fehlerbehebung: - CPU urlöschen; wenn Fehler noch vorhanden, - Spannung am AG ausschalten, CPU ziehen und wieder stecken und urlöschen; wenn Fehler noch vorhanden, - CPU austauschen bzw. reparieren lassen.

Melde- und Fehleranzeige-LEDs

LED "BASP"

Sie zeigt an, ob das S5-Bus-Signal BASP (Befehlsausgabe sperren) aktiv ist:
Im Einzelprozessor-Betrieb wird BASP von der CPU beim Eintritt in den Zustand RUN inaktiv und beim Übergang in den Zustand STOP aktiv geschaltet. Im ANLAUF und im STOP sowie im ersten Zyklus nach WIEDERANLAUF ist BASP aktiv gesetzt.
Im Mehrprozessor-Betrieb ist die Anzeige von BASP identisch mit der Anzeige im Einzelprozessorbetrieb, sofern der Schalter auf dem Koordinator sich in der Stellung RUN befindet. (Zum Sonderfall "Testbetrieb" lesen Sie bitte in Ihrem Systemhandbuch /2/ nach.)

Hinweis
Ist BASP aktiv, so sind alle digitalen Ausgänge gesperrt.

Ist vor dem Übergang in den Betriebszustand RUN ein AUTOMATISCHER oder MANUELLER WIEDERANLAUF durchgeführt worden, erlischt die BASP-LED erst dann, wenn der Restzyklus abgearbeitet worden ist.

LED "QVZ"

Quittungsverzug einer Peripheriebaugruppe.

LED "ADF"

Adressierfehler; das Anwenderprogramm hat eine Adresse im Prozeßabbild angesprochen, unter der in der Peripherie keine Baugruppe gesteckt ist.

LED "ZYK"

Zyklusfehler; Überschreitung der Zyklusüberwachungszeit.

Die Fehler ADF und QVZ können im ANLAUF und im RUN auftreten, der Zyklusfehler ZYK nur im RUN.

Am Ende einer Fehlerbearbeitungsebene ADF, QVZ oder ZYK wird die Fehler-LED vom Systemprogram gelöscht, falls nicht in den Zustand STOP verzweigt wurde.

4.2 Programmbearbeitungsebenen

Bild 4-2 gibt Ihnen eine Übersicht über die Betriebszustände und Bearbeitungsebenen bei der CPU 928B (-3UB12). Die Erklärungen zu den Abkürzungen in den Fehlerebenen finden Sie auf der nächsten Seite.

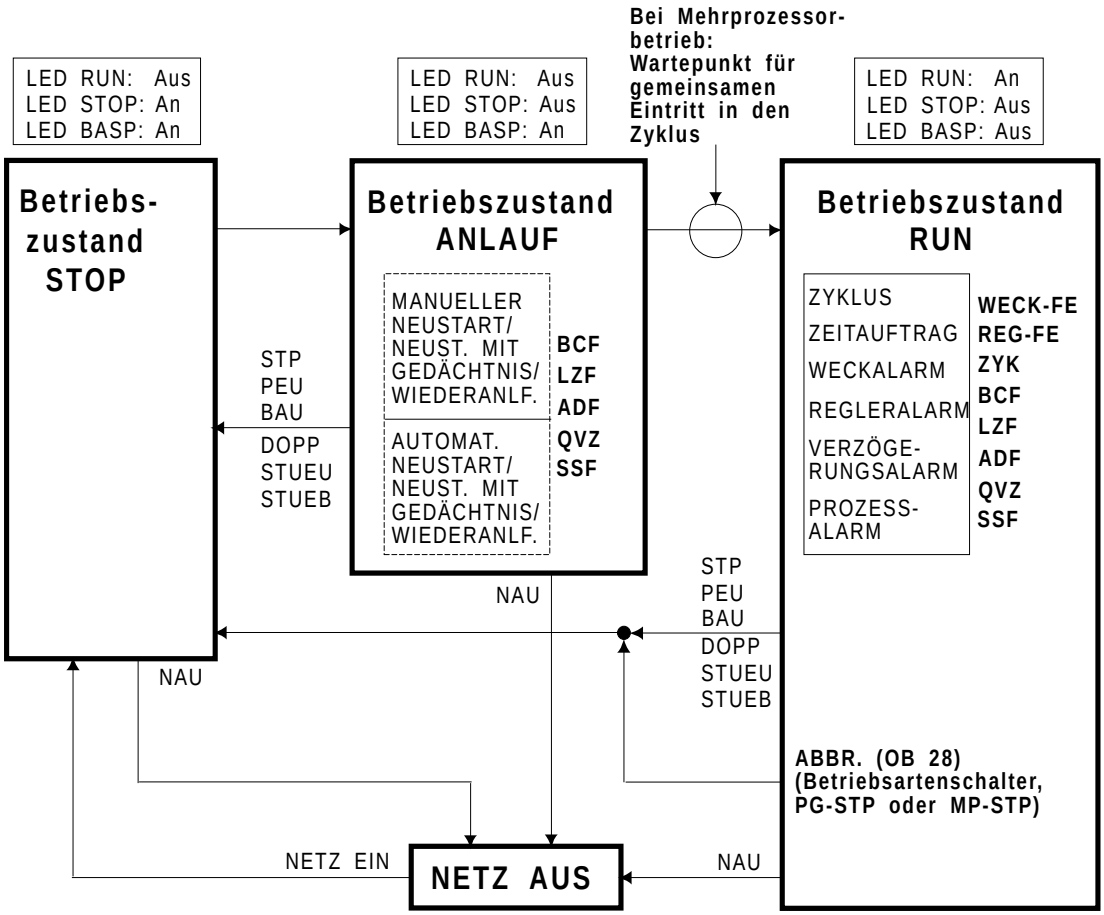


Bild 4-2 Betriebszustände und Programmbearbeitungsebenen

Programmbearbeitungsebenen im ANLAUF:

MANUELLER NEUSTART	Anlaufebenen
MANUELLER WIEDERANLAUF	
MANUELLER NEUSTART MIT GEDÄCHTNIS	
AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS	
AUTOMATISCHER NEUSTART	
AUTOMATISCHER WIEDERANLAUF	

BCF	(Befehlscodefehler)	Fehlerebenen
LZF	(Laufzeitfehler)	
ADF	(Adressierfehler)	
QVZ	(Quittungsverzug)	
SSF	(Schnittstellenfehler)	

Programmbearbeitungsebenen im RUN:

ZYKLUS		(zyklische Programmbearbeitung)	Grundebenen
ZEITAUFTRAG		(zeitgesteuerte Programmbearbeitung)	
WECKALARM	5 s	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	2 s	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	1 s	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	500 ms	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	200 ms	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	100 ms	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	50 ms	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	20 ms	(zeitgesteuerte Programmbearbeitung)	
WECKALARM	10 ms	(zeitgesteuerte Programmbearbeitung)	
REGLERALARM		(zeitgesteuerte Bearbeitung von Reglern) [1)]	
VERZÖGERUNGSALARM		(zeitgesteuerte Programmbearbeitung)	
PROZESSALARM		(alarmgesteuerte Programmbearbeitung)	

WECK-FE	(Weckfehler)	Fehlerebenen
REG-FE	(Reglerfehler)	
ZYK	(Zyklusfehler)	
BCF	(Befehlscodefehler)	
LZF	(Laufzeitfehler)	
ADF	(Adressierfehler)	
QVZ	(Quittungsverzug)	
SSF	(Schnittstellenfehler)	
ABBR	(Abbruch)	

Merkmale einer Programmbearbeitungsebene

Eine Programmbearbeitungsebene ist durch bestimmte Merkmale charakterisiert, die auf den folgenden Seiten erläutert werden.

[1)] ab Version -3UB12

Einschachtelung von anderen Ebenen

Wenn ein Ereignis eintritt, das eine höherpriore Bearbeitung erfordert, wird die aktuelle Ebene vom Systemprogramm unterbrochen und die höherpriore Ebene eingeschachtelt.

Die Einschachtelung erfolgt

- bei Fehlerebenen und Programmbearbeitungsebenen im ANLAUF: grundsätzlich an Befehlsgrenzen,
- bei allen anderen Ebenen: an Baustein- oder an Befehlsgrenzen (je nach DX-0-Einstellung – siehe Kapitel 7)

spezifisches Systemprogramm

Jede Programmbearbeitungsebene hat ihr spezisches Systemprogramm.

Beispiel:

In der Bearbeitungsebene ZYKLUS aktualisiert das Systemprogramm das Prozeßabbild der Ein- und Ausgänge, triggert die Zyklusüberwachungszeit und ruft die Verwaltung der PG-Schnittstelle auf (Systemkontrollpunkt).

USTACK

Nach Aufruf eines Organisationsbausteins durch das Systemprogramm führt die CPU die darin enthaltenen STEP-5-Anweisungen aus. Dabei wird der aktuelle Registersatz in den USTACK gerettet und ein **neuer Registersatz** angelegt (Register: AKKU 1 bis 4, Bausteinstack-Pointer, Baustein-Adreßregister, Datenbaustein-Anfangsadresse, Datenbaustein-Länge, STEP-Adreßzählerund das Basisadreßregister).
Ist durch das Auftreten des Ereignisses die "normale" Programmbearbeitung unterbrochen worden, so setzt die CPU nach der Bearbeitung des OB – inklusive aller dort eingeschachtelten Bausteine – die unterbrochene Programmbearbeitung an der Unterbrechungsstelle fort, sofern innerhalb des OB kein Stopp programmiert ist.

Beispiel:

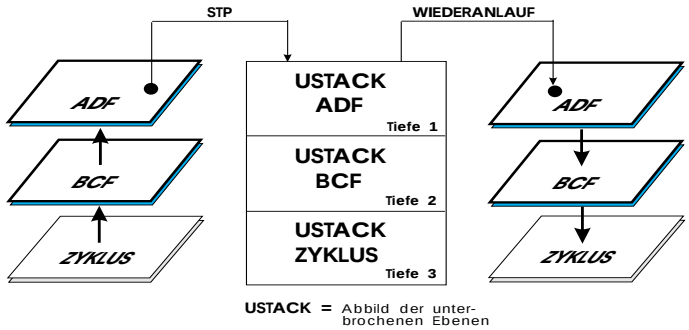
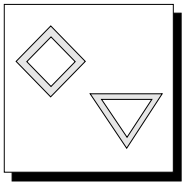


Bild 4-3 Prinzip des Ebenenwechsels und des USTACKs

Priorität



Die Programmbearbeitungsebenen haben eine feste Priorität. Davon abhängig können sie einander unterbrechen bzw. ineinander verschachtelt werden.

Die **"Anlauf- und die Fehlerebenen"** unterscheiden sich von den "Grundebenen" dadurch, daß sie grundsätzlich an der nächsten Befehlsgrenze eingeschachtelt werden, sobald das entsprechende Ereignis aufgetreten ist. Sie können sowohl in die Grundebenen als auch gegenseitig eingeschachtelt werden. Bei Fehlern hat der zuletzt aufgetretene immer die höchste Priorität.

Eine **"Grundebene"** hingegen kann in die jeweils niederpriore nur an Bausteingrenzen eingeschachtelt werden, es sei denn, diese Voreinstellung ist durch eine entsprechende Programmierung des DX 0 geändert worden (siehe Kapitel 7).

Priorität der "Grundebenen":

ZYKLUS
ZEITAUFTRAG
WECKALARM 5 s
WECKALARM 2 s
.
.
REGLERALARM
VERZÖGERUNGSALARM
PROZESSALARM

↓ aufsteigende Priorität

Beispiel:

Während der Bearbeitung eines Weckalarms tritt ein Prozeßalarm auf. Da der Prozeßalarm eine höhere Priorität besitzt, wird die Bearbeitung der Weckalarm-Ebene an der nächsten Bausteingrenze unterbrochen und die PROZESS-ALARM-Programmbearbearbeitungsebene eingeschachtelt. Wird bei der Bearbeitung des Prozeßalarms nun z.B. vom Systemprogramm ein Adressierfehler erkannt, so wird der Prozeßalarm sofort an der nächsten Befehlsgrenze unterbrochen, um die ADF-Ebene einzuschachteln.

Reaktion bei einem Doppelfehler

Eine einmal aktivierte Fehlerebene (ADF, BCF, LZF, QVZ, REG, ZYK), die noch nicht vollständig abgearbeitet ist, kann nicht noch einmal aktiviert werden, auch nicht, wenn eine andere Programmbearbeitungsebene dazwischengeschachtelt ist. **In diesem Fall geht die CPU wegen Doppelaufruf einer Programmbearbeitungsebene (im USTACK: DOPP) unmittelbar in STOP** (Ausnahme: Weckfehler, siehe dort). Im USTACK mit der Tiefe '01' ist die Kennung DOPP sowie die doppelt aufgerufene Fehlerebene angekreuzt.

Beispiele für Doppelfehler

Beispiel 1:

Bei der Bearbeitung der ADF-Ebene (AW-Schnittstelle OB 25) tritt ein weiterer Adressierfehler auf. Da die ADF-Ebene noch aktiviert ist, kann sie kein zweites Mal aufgerufen werden: Die CPU geht in STOP.

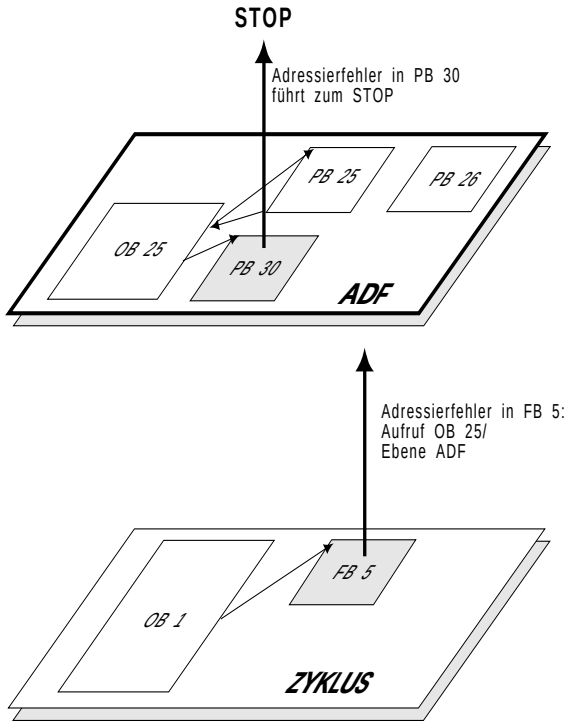


Bild 4-4 Ebenenwechsel bei einem Doppelfehler

Beispiel 2:

Bei Auftreten eines Operationscodefehlers in der LZF- Programmbearbeitungsebene versucht das Systemprogramm, die BCF-Ebene (AW-Schnittstelle OB 29) aufzurufen. Diese ist jedoch schon durch das Auftreten eines Parameterfehlers (AW- Schnittstelle OB 30) aktiviert worden und nicht vollständig abgearbeitet. Ein erneuter Aufruf der BCF-Ebene an dieser Stelle ist deshalb nicht zulässig: Die CPU geht in STOP (siehe Bild 4-5).

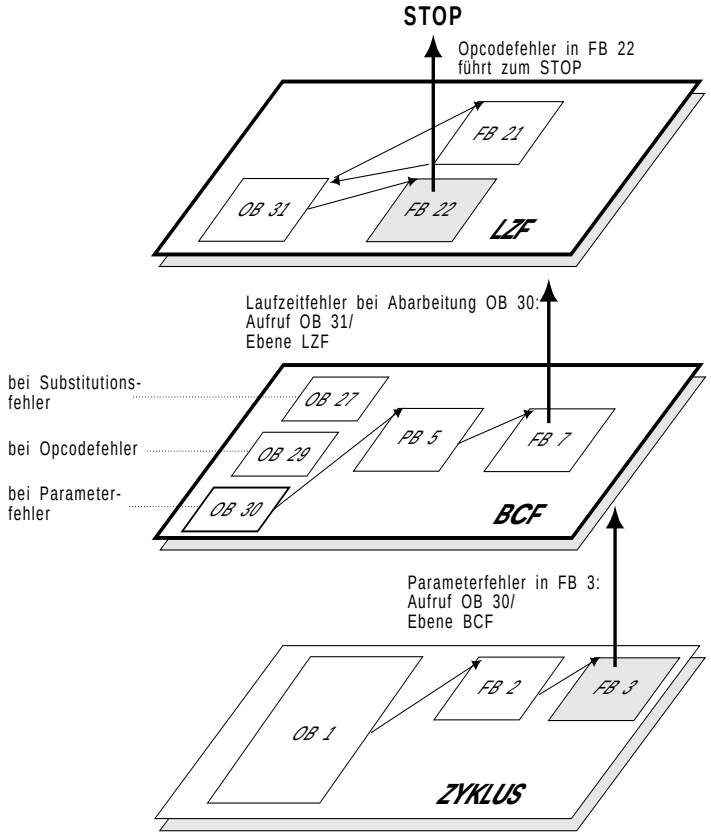


Bild 4-5 Doppelter Aufruf der Fehlerebene BCF

Beschreibung der einzelnen Ebenen

Die einzelnen Programmbearbeitungsebenen mit den dazugehörigen Anwenderschnittstellen werden in folgenden Abschnitten genauer beschrieben:

Abschnitt 4.4	beschreibt die Programbearbeitungsebenen im ANLAUF.
Abschnitt 4.5	beschreibt die Programmmbearbeitungsebenen im RUN.
Abschnitte 5.6 und 5.7	beschreiben die Fehlerebenen im ANLAUF und im RUN.

4.3 Betriebszustand STOP

4.3.1 Kennzeichen und Anzeige des Betriebszustandes

Der Betriebszustand STOP ist durch folgende Merkmale charakterisiert:

Anwenderprogramm

Es findet keine Anwenderprogrammbearbeitung statt.

Erhalt von Daten

Hat zuvor eine Programmbearbeitung stattgefunden, so bleiben die Werte von Zählern, Zeiten, Merkern und des Prozeßabbildes beim Übergang in den Stoppzustand erhalten.

BASP-Signal

Das Signal BASP (Befehlsausgabe sperren) ist aktiv. Damit sind alle digitalen Ausgaben gesperrt.

Ausnahme: Im Testbetrieb bei Mehrprozessorbetrieb wird BASP nicht aktiv – lesen Sie dazu bitte in Ihrem Systemhandbuch /2/ nach.

USTACK

Hat zuvor eine Programmbearbeitung stattgefunden, so ist im Stoppzustand für jede unterbrochene Programmbearbeitungsebene im Unterbrechungstack (USTACK) ein Informationsblock über die Unterbrechungsursachen vorhanden (siehe Abschnitt 5.4).

LEDs auf der Frontplatte der Zentralbaugruppe:

RUN-LED:	aus
STOP-LED:	an (Dauerlicht oder blinkend)
BASP-LED:	an (außer im Testbetrieb)

Die **STOP-LED** signalisiert Ihnen dabei, welche Ursachen für den momentanen Stoppzustand in Frage kommen. Dauerlicht oder Blinken der STOP-LED haben eine bestimmte Bedeutung, die im folgenden kurz beschrieben wird.

STOP-LED zeigt Dauerlicht

Der Betriebszustand STOP wurde ausgelöst:

- im Einzelprozessorbetrieb:
  - durch Betätigen des Betriebsartenschalters von RUN auf STOP,
  - durch PG-Funktion AG-STOP,
  - durch Gerätefehler (BAU, PEU),
  - nach URLÖSCHEN,
  - durch PG-Funktion "Bearbeitungskontrolle Ende".
- im Mehrprozessorbetrieb:
  - durch Betätigen des Betriebsartenschalters am KOR auf STOP,
  - durch STOP einer anderen CPU aufgrund einer Störung (eine **nicht fe**hlerverursachende CPU zeigt Dauerlicht).

STOP-LED blinkt langsam (ca 0,5 mal pro Sekunde)

Ein langsames Blinken der STOP-LED signalisiert Ihnen die unten aufgeführten Fälle. Im Mehrprozessorbetrieb wird durch langsames Blinken diejenige CPU gekennzeichnet, die den Stoppzustand (durch einen Fehler) verursacht hat.

Die STOP-LED blinkt langsam

- nach einem Stoppbefehl im Anwenderprogramm,
- bei Fehlbedienung (DB-1-Fehler, Wahl einer unzulässigen Anlaufart usw.),
- bei Programmier- oder Gerätefehlern (Aufruf eines nicht geladenen Bausteins, Adressierfehler, Quittungsverzug, Befehlscode-fehler usw.); als zusätzlicher Hinweis auf die Fehlerursache leuchten:
  LED "ADF"
  LED "QVZ"
  LED "ZYK"
- bei PG-Funktion "Bearbeitungskontrolle Ende" an dieser CPU.

STOP-LED blinkt schnell (ca 2 mal pro Sekunde)

Ein schnelles Blinken der STOP-LED signalisiert Ihnen die Warnung: "URLÖSCHEN" ist angefordert!

4.3.2 URLÖSCHEN anfordern

Anforderung vom Systemprogramm

Nach jedem Einschalten der Netzspannung und nach Durchführen des Urlöschens durchläuft die CPU eine Initialisierungsroutine. Wenn nun bei der Initialisierung Fehler festgestellt werden, die einen Programmstart nicht zulassen, geht die CPU in den Stoppzustand mit schnellem Blinken.

Mögliche Fehler:
- Inhalte der RAMs sind nicht korrekt.
  Abhilfe: CPU urlöschen.
- Inhalt des Anwender-EPROM ist nicht korrekt
  Abhilfe: Programmiertes EPROM stecken und CPU urlöschen.

Die Störungsursache muß beseitigt und danach die CPU (erneut) urgelöscht werden. URLÖSCHEN wird auch angefordert bei Auftreten eines CPU- oder Systemfehlers. Sie erkennen diese Fehler daran, daß die Aufforderung nach durchgeführtem URLÖSCHEN erneut auftritt. In diesem Fall wenden Sie sich bitte an Ihre nächste Siemens-Vertretung.

Anforderung vom Anwender

Durch folgende Bedienschritte fordern Sie URLÖSCHEN an:

1. Betätigen Sie den Betriebsartenschalter von RUN nach STOP.
   Ergebnis: Die CPU befindet sich im Stoppzustand. Die STOP-LED zeigt Dauerlicht.

2. Halten Sie den Betiebsarten**taster** in Stellung OVERALL RESET fest; gleichzeitig betätigen Sie den Betriebsarten**schalter** von STOP nach RUN und wieder nach STOP.
   Ergebnis: URLÖSCHEN ist angefordert. Die STOP-LED blinkt schnell.

Hinweis
Soll das angeforderte URLÖSCHEN doch nicht ausgeführt werden, so führen Sie jetzt einen NEUSTART oder MANUELLEN WIEDERANLAUF durch.

4.3.3 URLÖSCHEN durchführen

Unabhängig davon, ob die Urlöschanforderung vom Systemprogramm oder vom Anwender kommt, führen Sie das URLÖSCHEN folgendermaßen durch:

- Halten Sie den Betriebsarten**taster** in Stellung OVERALL RESET fest; gleichzeitig betätigen Sie den Betriebsarten**schalter** von STOP nach RUN und wieder nach STOP.
  Ergebnis: URLÖSCHEN wird durchgeführt. Die STOP-LED zeigt Dauerlicht.

- Oder: Durch die PG-Funktion URLÖSCHEN.
  (Beim URLÖSCHEN mit dem PG kann die manuelle Urlöschanforderung durch Schalterbetätigung entfallen! Die Stellungen von Betriebsartenschalter und -taster sind dann nicht relevant.)
  Ergebnis: URLÖSCHEN wird durchgeführt. Die STOP-LED zeigt Dauerlicht.

Hinweis
Haben Sie URLÖSCHEN durchgeführt, so ist als Anlaufart anschließend nur ein NEUSTART zulässig!

4.4 Betriebszustand ANLAUF

Der Betriebszustand ANLAUF ist durch folgende Merkmale charakterisiert:

Übergang von STOP in RUN

Der ANLAUF ist der Übergang vom Betriebszustand STOP in den Betriebszustand RUN.

Anlaufarten

Die CPU 928B kennt folgende Anlaufarten:

- NEUSTART (manuell oder automatisch),
- WIEDERANLAUF (manuell oder automatisch),
- NEUSTART MIT GEDÄCHTNIS (NSMG, manuell oder automatisch – nur bei Version -3UA12).

Nach einem **NEUSTART** wird das zyklische Anwenderprogramm von Anfang an bearbeitet.
Nach einem **WIEDERANLAUF** wird die zyklische Anwenderprogrammbearbeitung an der Unterbrechungsstelle fortgesetzt.

Organisationsbausteine

Folgende Organisationsbausteine werden aufgerufen:

bei MANUELLEM oder AUTOMATISCHEM NEUSTART:	OB 20
bei MANUELLEM WIEDERANLAUF bzw. NSMG	OB 21
bei AUTOMATISCHEM WIEDERANLAUF bzw. NSMG:	OB 22

Die Länge dieser Organisationsbausteine ist nicht beschränkt; ihr Ablauf wird zeitlich nicht überwacht. In den Anlauf-Organisationsbausteinen können weitere Bausteine aufgerufen werden.

Datenbehandlung

Die Werte von Zählern, Zeiten, Merkern und Prozeßabbildern werden in den einzelnen Anlaufarten unterschiedlich behandelt.

BASP-Signal

Das Signal BASP (Befehlsausgabe sperren) ist aktiv. Damit sind alle digitalen Ausgaben gesperrt.
Ausnahme: Im Testbetrieb wird BASP nicht aktiv gesetzt! (Zum Testbetrieb lesen Sie bitte in Ihrem Systemhandbuch nach.)

LEDs auf der Frontplatte der Zentralbaugruppe:

RUN-LED:	aus
STOP-LED:	aus
BASP-LED:	an (außer im Testbetrieb)

Anlaufverhalten im Mehrprozessorbetrieb

Hinweise zum "Anlaufverhalten im Mehrprozessorbetrieb" finden Sie in Kapitel 10.1.7.

4.4.1 MANUELLER und AUTOMATISCHER NEUSTART

Wann ist NEUSTART zulässig ?

Ein NEUSTART ist **immer zulässig**, sofern vom Systemprogramm nicht URLÖSCHEN angefordert ist.

MANUELLER NEUSTART

So **lösen Sie** den MANUELLEN NEUSTART aus:

- Halten Sie den Betriebsarten**taster** in Stellung RESET; betätigen Sie gleichzeitig den Betriebsartenschalter von STOP nach RUN.
- Oder: Benutzen Sie die PG-Funktion AG-START (NEUSTART).

AUTOMATISCHER NEUSTART

Ein AUTOMATISCHER NEUSTART **wird ausgelöst**:

Nach Netzspannungsausfall/NETZ AUS im ANLAUF oder RUN und anschließender Netzspannungswiederkehr/NETZ EIN durchläuft die CPU eine Initialisierungsroutine und versucht dann automatisch, einen NEUSTART durchzuführen, sofern der DX 0 entsprechend parametriert ist (siehe Kapitel 7).

Voraussetzungen:

- Die Schalter an allen CPUs und am Koordinator stehen unverändert auf RUN.
- Bei der Initialisierung sind keine Fehler aufgetreten.
- Die CPU befand sich vor dem Spannungsausfall/NETZ AUS nicht im STOP.

Bei Netzspannungsausfall in einem Erweiterungsgerät (PEU-Signal) geht die CPU in STOP. Sie verharrt dort solange, bis das PEU-Signal inaktiv geschaltet wird und versucht dann, einen AUTOMATISCHEN NEUSTART bzw. AUTOMATISCHEN WIEDERANLAUF durchzuführen.

4.4.2 MANUELLER und AUTOMATISCHER WIEDERANLAUF

*Wann ist ein WIEDERANLAUF **nicht** zulässig?*

Ein WIEDERANLAUF ist **nicht** zulässig,

- wenn vom Systemprogramm URLÖSCHEN angefordert ist

  oder

- nach einem der folgenden Ereignisse:
  - Doppelaufruf einer Programmbearbeitungsebene (USTACK: DOPP),
  - URLÖSCHEN (Steuerbits: URGELOE),
  - Anlaufabbruch (Steuerbits: ANL-ABB),
  - STOP nach PG-Funktion BEARBEITUNGSKONTROLLE ENDE,
  - Komprimieren im STOP,
  - Stacküberlauf,
  - Änderung des Anwenderprogramms im STOP.

MANUELLER WIEDERANLAUF

So lösen Sie einen MANUELLEN WIEDERANLAUF aus**:**

- Der Betriebsarten**taster** befindet sich in Mittelstellung.
- Betätigen Sie den Betriebsarten**schalter** von STOP nach RUN.
- Oder: Benutzen Sie die PG-Funktion AG-START (WIEDERANLAUF).

AUTOMATISCHER WIEDERANLAUF

Ein AUTOMATISCHER WIEDERANLAUF **wird ausgelöst**:

Nach Netzspannungsausfall/NETZ AUS im ANLAUF oder RUN und anschließender Netzspannungswiederkehr/NETZ EIN durchläuft die CPU eine Initialisierungsroutine und versucht dann automatisch, einen WIEDERANLAUF durchzuführen, sofern der DX 0 entsprechend parametriert ist (siehe Kapitel 7).

Voraussetzungen:

- Die Schalter an allen CPUs und am Koordinator stehen unverändert auf RUN.
- Bei der Initialisierung sind keine Fehler aufgetreten.
- Die CPU befand sich vor dem Spannungsausfall/NETZ AUS nicht im STOP.

Bei Netzspannungsausfall in einem Erweiterungsgerät (PEU-Signal) geht die CPU in STOP. Sie verharrt dort solange, bis das PEU-Signal inaktiv geschaltet wird und versucht dann, einen AUTOMATISCHEN WIEDERANLAUF bzw. AUTOMATISCHEN NEUSTART durchzuführen.

NEUSTART MIT GEDÄCHTNIS (ab Version -3UB12)

Ist im Datenbaustein DX 0 der Parameter "Neustart mit Gedächtnis" abgelegt, so führt das Systemprogramm an Stelle des WIEDERANLAUFS den NEUSTART MIT GEDÄCHTNIS durch. Wie sich dieser von einem "normalen" NEUSTART unterscheidet, entnehmen Sie bitte dem folgenden Abschnitt.

4.4.3 Gegenüberstellung der unterschiedlichen Anlaufarten

Tabelle 4-2 Gegenüberstellung der unterschiedlichen Anlaufarten

Systemprogramm-leistung	NEUSTART		WIEDERANLAUF		NEUSTART MIT GEDÄCHTNIS	
	manuell	automat.	manuell	automat.	manuell	automat.
Auswerten von:						
- DB 1	ja	ja	nein	nein	nein	nein
- DB 2	ja	ja	nein	nein	nein	nein
- DX 0	ja	ja	nein	nein	nein	nein
- DX 2	ja	ja	nein	nein	nein	nein
Initialisieren von:						
- DB 0	nein [1)]	nein [1)]	nein [1)]	nein [1)]	nein [1)]	nein [1)]
- 9. Spur	ja	ja	nein	nein	nein	nein
- Alarme sperren/verzögern	ja	ja	nein	nein	ja	ja
- Zyklusstatistik	ja	ja	nein	nein	nein	nein
Löschen von:						
- Uhrzeitauftrag	ja	ja	nein	nein	nein	nein
- Verzögerungsalarm	ja	ja	ja	ja	ja	ja
- USTACK/ BSTACK	ja	ja	nein	nein	ja	ja
- Prozeßabbild der Eingänge	ja (vollständig)	ja (vollständig)	nein	nein	nein	nein
- Prozeßabbild der Ausgänge/ digitale Periph.	ja (vollständig)	ja (vollständig)	nein	nein	ja (gemäß 9. Spur)	ja (gemäß 9. Spur)
- analoge Periph.	ja	ja	nein	nein	nein	nein

Systemprogramm-leistung	NEUSTART		WIEDERANLAUF		NEUSTART MIT GEDÄCHTNIS	
	manuell	automat.	manuell	automat.	manuell	automat.
Fortsetzung der Tabelle 4-2						
Löschen von (Forts):						
- Koppelmerker	ja	ja	nein	nein	nein	nein
- Semaphoren	ja	ja	nein	nein	nein	nein
- M- und S-Merkern	ja	ja	nein	nein	nein	nein
- Zeiten und Zählern	ja	ja	nein	nein	nein	nein
Restzyklusbearbeitung bei aktivem BASP-Signal	nein	nein	ja	ja	nein	nein
von OB 223 festgestellte Anlaufart	NEUSTART	NEUSTART	MANUEL. WA	AUTOMAT. WA	MANUEL. WA	AUTOMAT. WA
Anzeige der Anlaufart am PG in den USTACK-Steuerbits	NEUSTA	NEUSTA + AWA	MWA	AWA	ANL-6 + MWA	ANL-6 + AWA
Anwenderschnittstelle	OB 20	OB 20	OB 21	OB 22	OB 21	OB 22

[1] Der DB 0 wird immer nach NETZ EIN oder nach URLÖSCHEN initialisiert.

Definition der "9. Spur"

Die "9. Spur" ist eine Liste der beim letzten NEUSTART quittierenden Ein- und Ausgangsbytes im Prozeßabbildbereich. Ist der DB 1 programmiert und geladen, so enthält die "9. Spur" nach einem erfolgten NEUSTART nur die im DB 1 aufgeführten Ein- und Ausgangsbytes.

Ein Zugriff mit STEP-5-Befehlen auf die 9. Spur ist nicht möglich.

4.4.4 Anwenderschnittstellen für den Anlauf

Als Anwenderschnittstellen für die verschiedenen Anlaufarten dienen die Organisationsbausteine OB 20, OB 21 und OB 22. In diesen Bausteinen können Sie Ihr STEP-5-Programm für die jeweilige Anlaufart hinterlegen.

Sie können in den Anlauf-OBs z. B.

- Merker setzen,
- Zeiten starten (der Start wird vom Systemprogramm bis zum Eintritt in den RUN verzögert),
- den Datenverkehr der CPU mit Peripheriebaugruppen vorbereiten,
- die Synchronisierung von CPs durchführen.

OB 20

NEUSTART:

Wenn die CPU einen MANUELLEN oder AUTOMATISCHEN NEUSTART durchführt, wird vom Systemprogramm **einmal** der OB 20 aufgerufen. Sie können dort ein STEP-5-Programm hinterlegen, das vor Beginn der zyklischen Programmbearbeitung vorbereitende Schritte für einen **Neubeginn** der zyklischen Bearbeitung durchführt.

Nach der Bearbeitung des OB 20 beginnt die zyklische Programmbearbeitung durch Aufruf des OB 1 bzw. FB 0.
Ist der OB 20 nicht geladen, beginnt die CPU am Ende eines NEUSTARTS (nach den Systemleistungen) sofort mit der zyklischen Programmbearbeitung.

OB 21

MANUELLER WIEDERANLAUF oder MANUELLER NEUSTART MIT GEDÄCHTNIS:

Wenn die CPU einen MANUELLEN WIEDERANLAUF oder MANUELLEN NEUSTART MIT GEDÄCHTNIS durchführt, so ruft das Systemprogramm einmalig den OB 21 auf. Sie können dort ein STEP-5-Programm hinterlegen, das einmalig vor Wiederaufnahme der vorher im RUN unterbrochenen Programmbearbeitung bestimmte Tätigkeiten ausführt.

MANUELLER WIEDERANLAUF

Nach der Bearbeitung des OB 21 wird bei MANUELLEM WIEDERANLAUF die zyklische Programmbearbeitung an der Abbruchstelle mit der nächsten Anweisung fortgesetzt. Es gilt:

- Das BASP-Signal (Befehlsausgabe sperren) bleibt während der Bearbeitung des Restzyklus aktiv und wird erst mit Beginn des nächsten (vollständigen) Zyklus inaktiv.
- Das Prozeßabbild der Ausgänge wird am Ende des Restzyklus zurückgesetzt.

Ist der OB 21 nicht geladen, beginnt die CPU am Ende eines MANUELLEN WIEDERANLAUFs nach den Systemleistungen sofort an der Unterbrechungsstelle mit der Programmbearbeitung.

Hinweis
Die CPU registriert einen Netzspannungsausfall (NAU oder PEU) auch dann, wenn er im STOP auftritt. Wenn Sie danach einen MANUELLEN **WIEDERANLAUF** auslösen, so ruft die CPU vor dem OB 21 den **OB 22** zur Bearbeitung auf.
Lösen Sie stattdessen einen MANUELLEN **NEUSTART** aus, so wird die Vorgeschichte von der CPU ignoriert und der OB 22 **nicht** aufgerufen.

MANUELLER NEUSTART MIT GEDÄCHTNIS

Ist im Datenbaustein DX 0 der Parameter "NEUSTART MIT GEDÄCHTNIS" eingetragen, so führt das Systemprogramm nach Bearbeitung des OB 21 einen **NEUSTART MIT GEDÄCHTNIS** durch (die CPU setzt die Programmbearbeitung mit der **ersten STEP-5-Anweisung im OB 1** bzw. **FB 0** fort). Die Signalzustände der Merker, Koppelmerker, Semaphore und die Baustein-Adreßliste (DB 0) **bleiben erhalten**.

OB 22

AUTOMATISCHER WIEDERANLAUF oder AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS:

Wenn die CPU einen AUTOMATISCHEN WIEDERANLAUF oder AUTOMATISCHEN NEUSTART MIT GEDÄCHTNIS durchführt, so ruft das Systemprogramm einmalig den OB 22 auf. Sie können dort ein STEP-5-Programm hinterlegen, das einmalig vor Wiederaufnahme der vorher im RUN unterbrochenen Programmbearbeitung bestimmte Tätigkeiten ausführt.

AUTOMATISCHER WIEDERANLAUF

Nach Rückkehr der Versorgungsspannung führt die CPU die bereits erwähnten Systemfunktionen aus und versucht das unterbrochene Programm an der Unterbrechungsstelle fortzusetzen.

Vorher wird – falls geladen – der OB 22 aufgerufen. Nach der Bearbeitung des OB 22 wird die unterbrochene Programmbearbeitung an der Abbruchstelle mit der nächsten Anweisung fortgesetzt.

Nach einem Netzausfall mit anschließender Netzwiederkehr gilt:

- Das BASP-Signal (Befehlsausgabe sperren) ist während der Bearbeitung des Restzyklus aktiv und wird erst mit Beginn des ersten vollständigen Zyklus inaktiv gesetzt.
- Das Prozeßabbild der Ausgänge wird am Ende des Restzyklus zurückgesetzt.

AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS

Ist im Datenbaustein DX 0 der Parameter "Neustart mit Gedächtnis" eingetragen, so führt das Systemprogramm nach Bearbeitung des OB 22 einen **NEUSTART mit GEDÄCHTNIS** durch (die CPU setzt die Programmbearbeitung mit der **ersten STEP-5-Anweisung im OB 1** bzw. **FB 0** fort). Die Signalzustände der Merker, Koppelmerker, Semaphore und die Baustein-Adreßliste (DB 0) **bleiben erhalten.**

4.4.5 Unterbrechungen im ANLAUF

Ein Anlaufprogramm kann unterbrochen werden durch

- Netzspannungsausfall im Zentralgerät (NAU) oder im Erweiterungsgerät (PEU),
- Stoppschalter, Stopp-Befehl, MP-STP oder PG-STP

  oder

- Programm- und Gerätefehler (siehe Abschnitt 5.6).

Soll der unterbrochene ANLAUF anschließend mit einer der möglichen Anlaufarten fortgesetzt werden, so beachten Sie bitte die folgenden Punkte:

Netzspannungsausfall im ANLAUF

Bei **Netzspannungsausfall im ANLAUF** und anschließender Netzwiederkehr müssen die der folgenden Tabelle aufgeführten Fälle unterschieden werden:

Eingestellte Betriebsart: AUTOMATISCHER WIEDERANLAUF
Die CPU führt gerade einen **NEUSTART**(OB 20) durch: Nach Netzausfall und anschließender Netzwiederkehr wird der Organisationsbaustein OB 22 (AUTOMATISCHER WIEDERANLAUF) an der unterbrochenen Stelle im OB 20 eingeschachtelt.
Die CPU führt gerade einen **MANUELLEN WIEDERANLAUF** (OB 21) durch: Nach Netzausfall und anschließender Netzwiederkehr wird der Organisationsbaustein OB 22 (AUTOMATISCHER WIEDERANLAUF) an der unterbrochenen Stelle im OB 21 eingeschachtelt.
Die CPU führt gerade einen **AUTOMATISCHEN IEDERANLAUF** (OB 22) durch: Nach Netzausfall und anschließender Netzwiederkehr wird kein zweiter OB 22 eingeschachtelt: Der unterbrochene OB 22 wird nach Rückkehr der Netzspannung nicht fortgesetzt, sondern abgebrochen, und es wird stattdessen der neu aufgerufene OB 22 bearbeitet.
Eingestellte Betriebsart: AUTOMATISCHER NEUSTART
Die CPU führt gerade einen MANUELLEN oder AUTOMATISCHEN NEUSTART oder einen MANUELLEN WIEDERANLAUF durch: Nach Netzausfall und anschließender Netzwiederkehr wird der unterbrochene OB 20 bzw. OB 21 nicht fortgesetzt, sondern abgebrochen, und es wird stattdessen der neu aufgerufene OB 20 bearbeitet.

Die gleichen Regeln gelten für einen AUTOMATISCHEN WIEDERANLAUF nach einem PEU-Signal.

MANUELLER WIEDERANLAUF nach Abbruch eines ANLAUFs

Wenn die CPU aus einem beliebigen ANLAUF in den STOP gegangen ist (z. B. wegen Stoppschalter oder wegen ADF) und Sie anschließend einen MANUELLEN WIEDERANLAUF auslösen, so wird der unterbrochene ANLAUF an der Abbruchstelle fortgesetzt. Es wird kein OB 21 eingeschachtelt.

MANUELLER NEUSTART nach Abbruch eines ANLAUFs

Wenn die CPU aus einem beliebigen ANLAUF in den STOP gegangen ist und Sie anschließend einen MANUELLEN NEUSTART auslösen, so wird der unterbrochene ANLAUF abgebrochen und ein NEUSTART durchgeführt. Dabei wird – sofern er geladen ist – der OB 20 aufgerufen.

Abbruch von NEUSTART MIT GEDÄCHTNIS

Ein NEUSTART MIT GEDÄCHTNIS wird **abgebrochen** durch

- Netzspannungsausfall im Zentralgerät (NAU) oder im Erweiterungsgerät (PEU),
- Stoppschalter, Stopp-Befehl, MP-STP oder PG-STP

  oder

- Programm- und Gerätefehler (siehe Abschnitt 5.6).

Ein abgerochener NEUSTART MIT GEDÄCHTNIS wird beim Wiederanlauf **nicht** fortgeführt. Stattdessen wird ein **neuer** NEUSTART MIT GEDÄCHTNIS gestartet.

Bei der Wahl der Anlaufart bleibt die Vorgeschichte unberücksichtigt. Insbesondere gilt:

- Wird ein MANUELLER oder AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS durch NETZ AUS oder Netzspannungsausfall im Erweiterungsgerät abgebrochen, so erfolgt bei NETZ EIN immer ein neuer **AUTOMATISCHER** NEUSTART MIT GEDÄCHTNIS, wenn alle sonstigen Anlaufbedingungen erfüllt sind.
- Wird ein MANUELLER oder AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS durch eine der übrigen Abbruchursachen ausgelöst, so erfolgt ein neuer MANUELLER NEUSTART MIT GEDÄCHTNIS.

4.5 Betriebszustand RUN

Wenn die CPU einen ANLAUF fertig bearbeitet hat (und nur dann), geht sie in den Betriebszustand **RUN**. Dieser Betriebszustand ist durch folgende Merkmale charakterisiert:

Bearbeitung des Anwenderprogramms

Das Anwenderprogramm im OB 1 bzw. im FB 0 wird zyklisch bearbeitet, wobei zusätzlich alarmgesteuert weitere Programmteile eingeschachtelt werden können.

Zeiten, Zähler, Prozeßabbild

Alle im Programm gestarteten Zeiten und Zähler "laufen", das **Prozeßabbild wird zyklisch aktualisiert.**

BASP-Signal

Das Signal BASP (Befehlsausgabe sperren) ist inaktiv gesetzt. Damit sind alle digitalen Ausgaben freigegeben.

Koppelmerker

Die Koppelmerker – falls im DB 1 programmiert – werden zyklisch aktualisiert.

LEDs auf der Frontplatte der Zentralbaugruppe:

RUN-LED: an
STOP-LED: aus
BASP-LED: aus

Hinweis
Ist vor dem Übergang in den Betriebszustand RUN ein AUTOMATISCHER oder MANUELLER WIEDERANLAUF durchgeführt worden, erlischt die BASP-LED erst dann, wenn der Restzyklus abgearbeitet und das Prozeßabbild aktualisiert worden ist. Der Betriebszustand **RUN** wird **nur** über den Betriebszustand **ANLAUF** erreicht!

Programmbearbeitungsebenen

Im Betriebszustand RUN gibt es 14 Grundprogrammbearbeitungsebenen:

- **ZYKLUS:** Das Anwenderprogramm wird zyklisch bearbeitet.
- **ZEITAUFTRAG:** Das Anwenderprogramm wird in festen, von Ihnen vorgegebenen Zeitintervallen bzw . einmalig zu einem festen Zeitpunkt bearbeitet (Uhrzeitgesteuerter Weckalarm).

- **9 WECKALARME:** Das Anwenderprogramm wird in festen, vom System angebotenen Zeitintervallen bearbeitet.
- **REGLERALARM:** Es wird eine vorgegebene Anzahl an Reglern zeitgesteuert bearbeitet.
- **VERZÖGERUNGS-ALARM:** Das Anwenderprogramm wird nach Ablauf einer eingestellten Verzögerungszeit einmalig bearbeitet.
- **PROZESSALARM:** Das Anwenderprogramm wird alarmgesteuert bearbeitet.

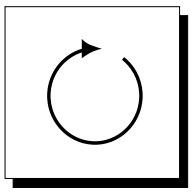
Die Bearbeitungsebenen unterscheiden sich in folgenden Punkten:

- Sie werden durch unterschiedliche Ereignisse ausgelöst.
- Für jede Programmbearbeitungsebene existiert als Anwenderschnittstelle ein anderer Organisationsbaustein bzw. Funktionsbaustein.

In einer CPU 928B können alle Grundbearbeitungsebenen gleichzeitig programmiert sein. Der Aufruf der entsprechenden Ebenen erfolgt entsprechend der vorgegebenen Priorität (siehe Kapitel 4.2) durch das Systemprogramm.

4.5.1 Zyklische Programmbearbeitung

Bei den speicherprogrammierbaren Steuerungen herrscht im allgemeinen **die zyklische Programmbearbeitung** (Programmbearbeitungsebene **ZYKLUS**) vor. Hierbei handelt es sich um einen "freien Zyklus", d. h. nach Erreichen des Zyklusendes beginnt sofort die nächste zyklische Bearbeitung (siehe Bild 4-6).



Auslösung

Hat die CPU das Anlaufprogramm fehlerfrei beendet, so beginnt sie mit der zyklischen Programmbearbeitung.

Prinzip

Prinzip der zyklischen Programmbearbeitung (Systemleistungen):

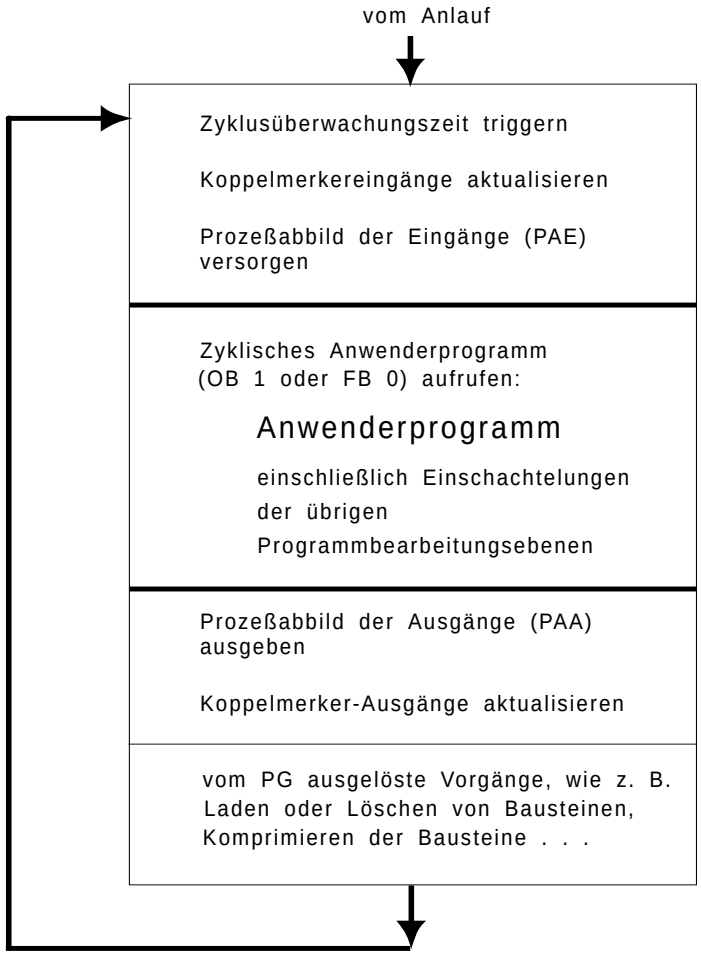


Bild 4-6 Zyklische Programmbearbeitung

Anwenderschnittstelle OB 1 bzw. FB 0

Bei der zyklischen Programmbearbeitung wird als Anwenderschnittstelle regelmäßig der Organisationsbaustein OB 1 oder der Funktionsbaustein FB 0 aufgerufen. Das STEP-5-Anwenderprogramm im OB 1 oder FB 0 wird von Anfang an über die darin von Ihnen programmierten Bausteinaufrufe hinweg durchgehend bearbeitet. Nach den Systemleistungen beginnt die CPU wieder von vorne mit der ersten STEP-5-Anweisung im OB 1 (bzw. im FB 0).

Im OB 1 programmieren Sie die Aufrufe der Programm-, Funktions- und Schrittbausteine, die im zyklischen Programm bearbeitet werden sollen.

Wenn Sie ein kurzes, zeitkritisches Anwenderprogramm haben, bei dem Sie auf eine strukturierte Programmierung verzichten können, programmieren Sie den FB 0: Da er über den gesamten Operationsvorrat von STEP 5 verfügt, können Sie Bausteinaufrufe einsparen und dadurch die Laufzeit des Programms verkürzen.

Hinweis
Wenn sowohl OB 1 als auch FB 0 in der CPU geladen sind, wird nur der OB 1 vom Systemprogramm aufgerufen. Wenn Sie den FB 0 als Anwenderschnittstelle verwenden, darf dieser keine Parameter enthalten!

Unterbrechungsstellen

Die zyklische Programmbearbeitung kann an **Bausteingrenzen** unterbrochen werden durch:

- eine prozeßalarmgesteuerte Programmbearbeitung,
- eine Reglerbearbeitung,
- eine zeitgesteuerte Programmbearbeitung.

Hinweis
Durch Parametrierung des DX 0 können diese Unterbrechungen auch an Befehlsgrenzen erfolgen - siehe Kapitel 7.

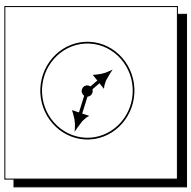
Die zyklische Programmbearbeitung kann an **Befehlsgrenzen** unterbrochen bzw. ganz abgebrochen werden

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch Stopp-Befehl.

AKKUs als Datenspeicher

Die Rechenregister AKKU 1, 2, 3 und 4 können bei der CPU 928B über Zyklusgrenzen hinweg – vom Ende eines Programmzyklus bis zum Beginn des nächsten – als Datenspeicher verwendet werden.

4.5.2 Zeitgesteuerte Programmbearbeitung



Eine zeitgesteuerte Bearbeitung liegt vor, wenn ein von einer Uhr oder von einem internen Takt kommendes Zeitsignal die CPU veranlaßt, die aktuelle Programmbearbeitung zu unterbrechen und ein spezifisches Programm zu bearbeiten. Nach der Bearbeitung dieses Programms kehrt die CPU zur Unterbrechungsstelle im unterbrochenem Programm zurück und setzt dort die Bearbeitung fort. Dadurch werden bestimmte Programmabschnitte zu einer gewünschten Zeit automatisch in die zyklische Programmbearbeitung eingeschoben.

Bei der zeitgesteuerten Programmbearbeitung stehen Ihnen unterschiedliche Arten der Auslösung zur Verfügung:

- Auslösung einmalig nach einer freigewählten Verzögerungszeit im Millisekunden-Bereich, ein **"Verzögerungsalarm"** (Programmbearbeitungsebene VERZÖGERUNGSALARM). Über diesen Alarm wird der Organisationsbaustein OB 6 aufgerufen.

- Auslösung in einem freigewähltem Zeitraster oder auch einmalig zu einem absoluten Zeitpunkt, ein **"uhrzeitgesteuerter Weckalarm"** (Programmbearbeitungsebene ZEITAUFTRAG). Über diesen Alarm wird der Organisationsbaustein OB 9 aufgerufen.

- Auslösung in 9 verschiedenen Zeitrastern im Bereich von 10 ms bis zu 5 Sekunden durch **"Weckalarme"** (Programmbearbeitungsebenen WECKALARME). Jedem Weckalarm ist ein Organisationsbaustein zugeordnet (OB 10 bis OB 18). Dabei handelt es sich um feste Zyklen, d. h. die Zeitdauer zwischen zwei Programmstarts ist fest.

Verzögerungsalarm (ab Version -3UB12)

Mit dem Verzögerungsalarm der CPU 928B lassen sich auch kleine Zeitintervalle mit einer Auflösung von 1 ms vorgeben. Wenn die eingestellte Zeit abgelaufen ist, so ruft das Systemprogramm **einmal** den OB 6 auf.

Auslösung

Ein Verzögerungsalarm wird durch Aufruf des Sonderfunktions-Organisationsbausteins OB 153 erzeugt (siehe Abschnitt 6.14). Sobald die mit dem OB 153 parametrierte Verzögerungszeit abgelaufen ist, unterbricht das Systemprogramm die laufende Programmbearbeitung und ruft den OB 6 auf. Danach wird die Programmbearbeitung an der Unterbrechungsstelle wieder aufgenommen.

Anwenderschnittstelle OB 6

Als Anwenderschnittstelle wird bei einem Verzögerungsalarm der OB 6 aufgerufen. Im OB 6 hinterlegen Sie ein STEP-5-Programm, das in diesem Fall bearbeitet werden soll. Ist der OB 6 nicht geladen, so wird die Programmbearbeitung nicht unterbrochen.

Unterbrechungen

Die Bearbeitung des Verzögerungsalarms kann an **Bausteingrenzen** oder an Befehlsgrenzen (Parametrierung DX 0) unterbrochen werden durch:

- die Bearbeitung eines Prozeßalarms.

Die Bearbeitung kann an **Befehlsgrenzen** unterbrochen bzw. ganz abgebrochen werden:

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch Stopp-Befehl.

Besonderheiten

- Ein Verzögerungsalarm wird nur im Betriebszustand RUN bearbeitet.
- Ein einmal erzeugter Verzögerungsalarm (= OB 153-Aufruf wurde bearbeitet) bleibt beim Übergang in den STOP und bei NETZ AUS **nicht** erhalten.
- Ein Verzögerungsalarm kann im ANLAUF und im RUN erzeugt werden (Aufruf des OB 153).
- Wenn Sie einen neuen Verzögerungsalarm erzeugen, d. h. den OB 153 mit neuen Parametern aufrufen, wird ein zuvor eingestellter Verzögerungsalarm storniert. Ein sich bereits in Bearbeitung befindender Verzögerungsalarm (OB 6 ist aktiv) wird fortgesetzt. Es ist also immer nur **ein** Verzögerungsalarm gültig.
- Tritt ein Verzögerungsalarm auf, ohne daß der vorangegangene vollständig bearbeitet ist, so wird der neue Alarm verworfen. **Verzögerungsalarme werden nicht auf Weckfehler überprüft!**
- Mit den Sonderfunktionen OB 120 und OB 122 können Sie die Bearbeitung von Verzögerungsalarmen sperren bzw. verzögern.

.

Uhrzeitgesteuerter Weckalarm

Auf der CPU 928B steht eine battterie-gepufferte Hardware-Uhr (zentrale Pufferung über die Stromversorgung des Zentralgerätes) zur Verfügung, die Sie per STEP-5-Programm setzen und auslesen können. Über diese Uhr läßt sich ein Programmteil zeitgesteuert bearbeiten.

Während der Verzögerungsalarm für schnelle Vorgänge eingesetzt wird, ist der uhrzeitgesteuerte Weckalarm besonders geeignet für die Bearbeitung von **einmaligen** Vorgängen oder Vorgängen, die in **großen Zeitabständen zyklisch** auftreten wie z. B. stündlich, täglich oder an jedem Montag. Ist der eingestellte Zeitpunkt erreicht, so ruft das Systemprogramm den OB 9 auf.

Auslösung

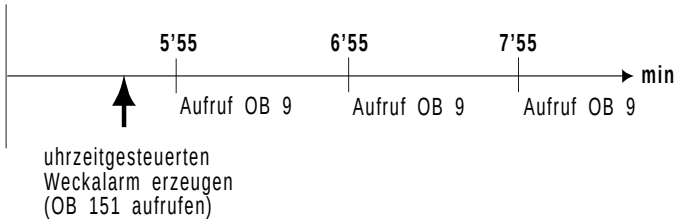
Ein uhrzeitgesteuerter Weckalarm (Zeitauftrag) wird durch Aufruf des Sonderfunktions-Organisationsbausteins OB 151 erzeugt (siehe Abschnitt 6.10). Ist der dem OB 151 übergebene Zeitpunkt (eine Uhrzeit, ein Datum) erreicht, so wird der Zeitauftrag bearbeitet. Dies kann einmalig sein (Absolutzeit) oder sich wiederholen (Zeitraster). Sobald ein Auftrag eintrifft, unterbricht das Systemprogramm die laufende Programmbearbeitung und ruft den OB 9 (Programmbearbeitungsebene ZEITAUFTRAG) auf. Danach wird die Programmbearbeitung an der Unterbrechungsstelle wieder aufgenommen.

Beispiel:

Es soll jede Minute beim Sekundenwert "55" ein Weckalarm ausgelöst werden.

Einstellung über den OB 151:

SEKUNDEN: **55**
AUFTAGSART **1** (minütlich)



Anwenderschnittstelle OB 9

Als Anwenderschnittstelle wird bei einem uhrzeitgesteuerten Weckalarm der OB 9 aufgerufen. Im OB 9 hinterlegen Sie ein STEP-5-Programm, das in diesem Fall bearbeitet werden soll. Ist der OB 9 nicht geladen, so wird die Programmbearbeitung nicht unterbrochen.

Unterbrechungen

Die Bearbeitung des uhrzeitgesteuerten Weckalarms kann an **Bausteingrenzen** oder an Befehlsgrenzen (Parametrierung DX 0) unterbrochen werden durch:

- die Bearbeitung eines Prozeßalarms,
- die Bearbeitung eines Verzögerungsalarms,
- die Bearbeitung eines Regleralarms.

Die Bearbeitung kann an **Befehlsgrenzen** unterbrochen bzw. ganz abgebrochen werden:

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch Stopp-Befehl STP.

Besonderheiten

- Ein uhrzeitgesteuerter Weckalarm wird nur im Betriebszustand RUN bearbeitet. Uhrzeitgesteuerte Weckalarme, die im STOP, bei **Netzausfall** oder im ANLAUF auftreten, werden **verworfen.**
- Ein nach URLÖSCHEN und NEUSTART erzeugter uhrzeitgesteuerter Weckalarm (= OB 151-Aufruf) bleibt bei einem WIEDERANLAUF und über NETZ AUS/NETZ EIN erhalten.
- Wenn Sie einen neuen uhrzeitgesteuerten Weckalarm erzeugen, d. h. den OB 151 mit neuen Zeitwerten aufrufen, wird ein zuvor eingestellter uhrzeitgesteuerter Weckalarm storniert. Ein sich gerade in Bearbeitung befindender uhrzeitgesteuerter Weckalarm wird fortgesetzt. Es ist also immer nur **ein** uhrzeitgesteuerter Weckalarm gültig.
- Tritt ein uhrzeitgesteuerter Weckalarm auf, ohne daß der vorangegangene vollständig bearbeitet ist, so wird der neue Weckalarm verworfen. **Uhrzeitgesteuerte Weckalarme werden nicht auf Weckfehler überprüft!**
- Mit den Sonderfunktionen OB 120 und OB 122 können Sie die Bearbeitung von Verzögerungsalarmen sperren bzw. verzögern.

Weckalarme

Programmbearbeitung in festen Zeitrastern:

In der CPU 928B können Sie bis zu 9 verschiedene Programme zeitgesteuert bearbeiten lassen, von denen jedes in einem anderen Zeitraster aufgerufen wird.

Auslösung

Ein Weckalarm wird automatisch in seinem Zeitraster ausgelöst, wenn der entsprechende OB geladen ist.

Anwenderschnittstellen

Als Anwenderschnittstelle wird bei Auftreten eines bestimmten Weckalarms an der nächsten Bausteingrenze (bzw. Befehlsgrenze) der zugehörige Organisationsbaustein eingeschachtelt.

Zuordnung der Weckalarmzeiten zu den OBs:

Tabelle 4-3 Zuordnung "Weckalarmzeit – aufgerufener OB"

Zeitraster	aufgerufener OB	
10 ms	OB 10	fallende
20 ms	OB 11	Priorität
50 ms	OB 12	↓
100 ms	OB 13	
200 ms	OB 14	
500 ms	OB 15	
1 s	OB 16	
2 s	OB 17	
5 s	OB 18	

Programmieren Sie beispielsweise im OB 13 denjenigen Programmteil, der alle 100 ms in die zyklische Programmbearbeitung eingeschoben werden soll.

Hinweis
Organisationsbausteine mit kürzeren Zeitrastern sind höherprior und können Organisationsbausteine mit längeren Zeitrastern unterbrechen!

Zeit seit letzter Alarmbearbeitung

Bei jedem Aufruf eines Weckalarm-OBs (OB 10 bis OB 18) wird im AKKU 1 hinterlegt, wieviele Zeitraster seit dem letzten Aufruf des Weckalarm-OBs aufgetreten sind. Dabei gilt:

AKKU 1 := Anzahl der Zeitraster - 1

Steht bei Aufruf des OB 11 beispielsweise die Zahl "5" im AKKU 1, bedeutet dies, daß seit dem letzten Aufruf des OB11 120 ms (6 Zeitraster) vergangen sind. Solange kein Weckfehler vorliegt, wird im AKKU 1 eine "0" übergeben.

Unterbrechungen

Die Bearbeitung eines Weckalarms kann entweder an **Bausteingrenzen** (Voreinstellung) oder an **Befehlsgrenzen** (Programmierung DX 0) unterbrochen werden durch

- die Bearbeitung eines Prozeßalarms,
- die Bearbeitung eines Verzögerungsalarms,
- die Bearbeitung eines Regleralarms,
- eine erneute Bearbeitung eines Weckalarms.

Die Bearbeitung kann an **Befehlsgrenzen** unterbrochen bzw. ganz abgebrochen werden:

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch Stopp-Befehl STP.

Hinweis
Eine zeitgesteuerte Programmbearbeitung kann nicht durch **dieselbe zeitgesteuerte** Programmbearbeitung unterbrochen werden (Weckfehler)!

Weckfehler (WECK-FE)

Wenn ein bestimmter Weckalarm-OB noch nicht vollständig bearbeitet ist und zum zweiten Mal bearbeitet werden soll, liegt ein Weckfehler vor. Ebenso kommt es zu einem Weckfehler, wenn ein bestimmter OB zum zweiten Mal aufgerufen wird, ohne daß der erste Aufruf bearbeitet worden wäre. Dies ist möglich bei der Einstellung "Weckalarme unterbrechen an Bausteingrenzen", besonders dann, wenn Ihr STEP-5-Programm langlaufende Bausteine enhält.
Liegt ein Weckfehler vor, so wird die Fehlerprogrammbearbeitungsebene WECK-FE aktiviert und das Systemprogramm ruft als Anwenderschnittstelle den **OB 33** auf. Im OB 33 können Sie die gewünschte Reaktion auf diesen Zustand programmieren.

Ist der OB 33 **nicht geladen**, geht die CPU bei Auftreten eines Fehlers in den Stoppzustand. Dann ist am Programmiergerät bei "Ausgabe USTACK" in den Steuerbits WECK-FE angekreuzt, im USTACK ist die Ebenenkennung des entsprechenden Weckalarms (EBENE) angegeben.

Beim Aufruf des OB 33 hinterlegt das Systemprogramm in AKU 1 und AKKU 2 zusätzliche Informationen, die den ersten aufgetretenen Fehler näher erläutern:

Tabelle 4-4 Weckfehlerkennungen

Fehlerkennung		Erläuterung
AKKU-1-L	**AKKU-2-L**	
1001H	0016H	Weckfehler bei OB 10 (10 ms)
1001H	0014H	Weckfehler bei OB 11 (20 ms)
1001H	0012H	Weckfehler bei OB 12 (50 ms)
1001H	0010H	Weckfehler bei OB 13 (100 ms)
1001H	000EH	Weckfehler bei OB 14 (200 ms)
1001H	000CH	Weckfehler bei OB 15 (500 ms)
1001H	000AH	Weckfehler bei OB 16 (1 sec)
1001H	0008H	Weckfehler bei OB 17 (2 sec)
1001H	0006H	Weckfehler bei OB 18 (5 sec)

Die Kennung im AKKU-2-L ist die Ebenenkennung (siehe Abschnitt 5.4) des fehlererzeugenden Weckalarms.

Programmbearbeitung fortsetzen

Soll die Programmbearbeitung bei einem aufgetretenen Weckfehler fortgesetzt werden, programmieren Sie entweder im OB 33 die Bausteinende-Anweisung 'BE', oder Sie ändern die Voreinstellung im DX 0 dahingehend, daß bei einem aufgetretenen Weckfehler und nicht programmiertem OB 33 die Programmbearbeitung trotzdem fortgesetzt wird.
Nach der Bearbeitung des OB 33 wird das Programm an der Unterbrechungsstelle fortgesetzt.

Hinweis
Beachten Sie im Hinblick auf die zeitgesteuerte Programmbearbeitung die Sonderfunktionen **OB 120, OB 121, OB 122** und **OB 123**, mit denen Sie die Bearbeitung von Weckalarmen für einen bestimmten Programmteil sperren bzw. verzögern können. (Dies ist möglich entweder für alle programmierten Weckalarme oder für einzelne von ihnen.)

Je "schneller" eine zeitgesteuerte Programmbearbeitungsebene ist, umso größer wird die Gefahr von Weckfehlern: Weckalarme mit kurzen Zeitrastern (z.B. der 10-ms- und der 20-ms-Weckalarm) werden im allgemeinen erfordern, daß sie auf Unterbrechung an Befehlsgrenzen eingestellt sind. Dies bedingt, daß auch der Regleralarm und der Prozeßalarm auf Unterbrechung an Befehlsgrenzen eingestellt sind (siehe Kapitel 7, DX-0-Parametrierung).

4.5.3 Regleralarm: Bearbeitung von Reglern

In der CPU 928B ist neben der zyklischen, der zeit- und alarmgesteuerten Programmbearbeitung zusätzlich auch die Bearbeitung von Reglern möglich. In vom Anwender festgelegten Zeitabschnitten (= Abtastzeit) wird die zyklische oder zeitgesteuerte Programmbearbeitung unterbrochen und der jeweilige Regler bearbeitet. Danach kehrt die CPU zur Unterbrechungsstelle im zyklischen oder zeitgesteuerten Programm zurück und setzt dort die Bearbeitung fort.

Auslösung

Ein Regleralarm wird nach Ablauf der vom Anwender gewählten Abtastzeit ausgelöst.

Leistungen des Systemprogramms:

- Es verwaltet die Anwenderschnittstelle für die Reglerbearbeitung.
- Es aktualisiert das Reglerprozeßabbild.

Anwenderschnittstelle: Standard-Funktionsbaustein "Reglerstruktur R64"

Bei der Reglerbearbeitung wird als Anwenderschnittstelle der R64-Standard-Funktionsbaustein aufgerufen. Dieser ermöglicht in Zusammenhang mit dem Regler-Parametrierungs-Baustein DB 2 die Bearbeitung von bis zu 64 Reglern.
Für jeden Regler parametrieren Sie einen bestimmten Datenbaustein. Im Datenbaustein DB 2, der sogenannten "Reglerliste", legen Sie fest, welche Regler zu welchem Zeitpunkt vom Systemprogramm zu bearbeiten sind. Der DB 2 ist für diese Aufgabe reserviert.

(Bei der Parametrierung, der Inbetriebnahme und beim Test des R64-Standard-FBs werden Sie unterstützt durch ein spezielles Programmpaket: "COMREG", siehe Katalog ST 59 /9/.)

Unterbrechungen

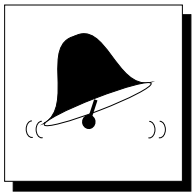
Eine Reglerbearbeitung kann entweder an **Bausteingrenzen** (Voreinstellung) oder an **Befehlsgrenzen** (Programmierung DX 0) unterbrochen werden durch

- die Bearbeitung eines Prozeßalarms,
- die Bearbeitung eines Verzögerungsalarms.

Die Bearbeitung kann an **Befehlsgrenzen** unterbrochen bzw. ganz abgebrochen werden:

- beim Auftreten eines Geräte- oder Programmfehlers,
- durch Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- durch Stopp-Befehl STP.

4.5.4 Prozeßalarm: Alarmgesteuerte Programmbearbeitung



Eine alarmgesteuerte Programmbearbeitung liegt vor, wenn ein S5-Bus-Signal einer interruptfähigen Digitaleingabebaugruppe (z. B. 6ES5 432-4UAxx) oder einer entsprechend arbeitenden IP-Baugruppe die CPU veranlaßt, die Programmbearbeitung zu unterbrechen und einen spezifischen Programmteil zu bearbeiten. Nach der Bearbeitung dieses Programms kehrt die CPU zur Unterbrechungsstelle zurück und setzt dort die Bearbeitung fort.

Die Auswertung des Prozeßalarms kann pegel- oder flankengetriggert erfolgen. Sie läßt sich per Programm sperren, verzögern oder freigeben. Der OB 2 kann wahlweise an **Befehls-** oder **Bausteingrenzen** (Programmierung DX 0) die laufende Programmbearbeitung unterbrechen.

Auslösung

Der aktive Zustand einer Interruptleitung auf dem S5-Bus löst den Prozeßalarm aus. Abhängig vom Steckplatz ist jeder CPU jeweils eine der Interruptleitungen zugeordnet (welche, entnehmen Sie bitte dem Systemhandbuch, Kapitel 4).

Anwenderschnittstelle: OB 2

Als Anwenderschnittstelle wird bei Auftreten eines Prozeßalarms der OB 2 aufgerufen. Im OB 2 programmieren Sie ein spezifisches Programm, das im Falle eines Prozeßalarms bearbeitet werden soll.

Ist der OB 2 **nicht geladen**, wird die Programmbearbeitung **nicht unterbrochen**. Es findet **keine** alarmgesteuerte Programmbearbeitung statt.

Unterbrechungen

Eine prozeßalarmgesteuerte Programmbearbeitung kann **nur** unterbrochen werden durch

- einen Programm- oder Gerätefehler (an Befehlsgrenzen),
- Bedienung (PG-Funktion, Stoppschalter, MP-STP),
- Stopp-Befehl.

Hinweis
Eine alarmgesteuerte Programmbearbeitung kann nicht durch eine **zeitgesteuerte** Programmbearbeitung oder eine **erneute alarmgesteuerte** Programmbearbeitung unterbrochen werden!

Mehrfachalarme

Treten während der alarmgesteuerten Programmbearbeitung erneut Prozeßalarme auf, so werden diese solange **ignoriert**, bis der OB 2 **vollständig bearbeitet ist** (inkl. aller im OB 2 aufgerufenen Bausteine). Dann kehrt die CPU an die Unterbrechungsstelle zurück und bearbeitet das Programm bis zur nächsten Bausteingrenze. Erst dann wird ein neuer Prozeßalarm akzeptiert und erneut der OB 2 aufgerufen. Dadurch wird auch bei einem Daueralarm das zyklische Programm bearbeitet. (Dies gilt nicht für den Fall, daß im DX 0 "Prozeßalarm unterbricht an Befehlsgrenzen" eingestellt ist).

Hinweis
Mehrfache Alarme werden nicht erkannt.

Der OB 2 wird auch dann aufgerufen, wenn bei Erreichen der Bausteingrenze im unterbrochenen Programm der Signalzustand der Interruptleitung bereits wieder passiv ist .

Prozeßalarme, die während der Bearbeitung des OB 2 auftreten und kürzer anstehen als die Bearbeitung des OB 2 dauert (wenn pegelgetriggert), werden nicht erkannt.
Der Signalzustand des Interrupt-Signals zwischen dem ersten Aktivwerden und dem Abschluß von OB 2 (BE-Befehl) ist irrelevant.

pegelgetriggerte Prozeßalarmsignale

In der Voreinstellung (DX 0) ist das Prozeßalarmsignal bei der CPU 928B pegelgetriggert, d. h. der aktive Zustand der Interruptleitung setzt eine Anforderung, die an der nächsten Baustein- oder Befehlsgrenze (je nach DX-0-Einstelung) zur Bearbeitung des OB 2 führt.

Ein Prozeßalarm wird auch dann erkannt und bearbeitet, wenn das Interrupt-Signal an der Bausteingrenze nicht mehr aktiv ist.

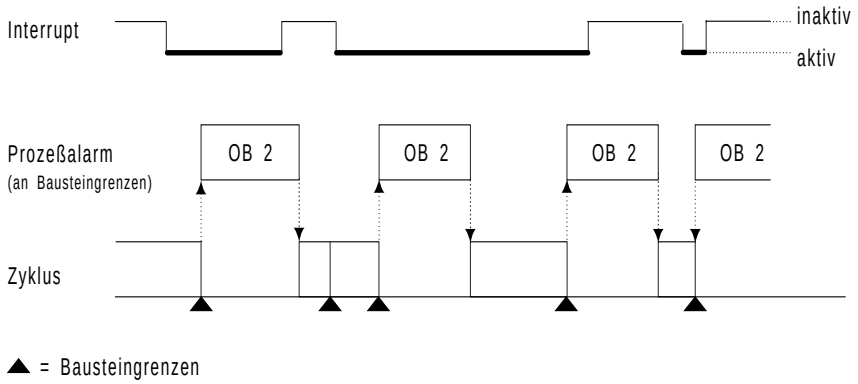


Bild 4-7 pegelgetriggerte Prozeßalarmsignale

Der aufgerufene OB 2 wird vollständig bearbeitet. Ist am Ende des OB 2 das Interruptsignal immer noch bzw. erneut aktiv, so wird im zyklischen Programm ein Baustein bearbeitet (nur bei Unterbrechung an Bausteingrenzen) und anschließend wieder der OB 2 aufgerufen. Steht der Pegel nicht mehr an, wird erst beim nächsten Signalzustandswechsel (von inaktiv nach aktiv) erneut der OB 2 aufgerufen.

Aktive Zustände des Interrupt-Signals **vor** dem Bearbeiten des Bausteinende-Befehls (BE) von OB 2 sind irrelevant.

flankengetriggerte Prozeßalarmsignale

Diese Einstellung erreichen Sie über die Parametrierung des DX 0. Nach Abarbeitung des OB 2 kann ein neuer Prozeßalarm nur durch einen Signalzustandswechsel (von inaktiv nach aktiv) ausgelöst werden. **Nach** dem Bearbeiten des Bausteinende-Befehls (BE) von OB 2 muß ein "**inaktiv-aktiv-Signalwechsel**" des Interrupt-Signals folgen, um einen Prozeßalarm zu erzeugen.

Ein Prozeßalarm wird auch dann erkannt und bearbeitet, wenn das Interrupt-Signal an der Bausteingrenze nicht mehr aktiv ist.

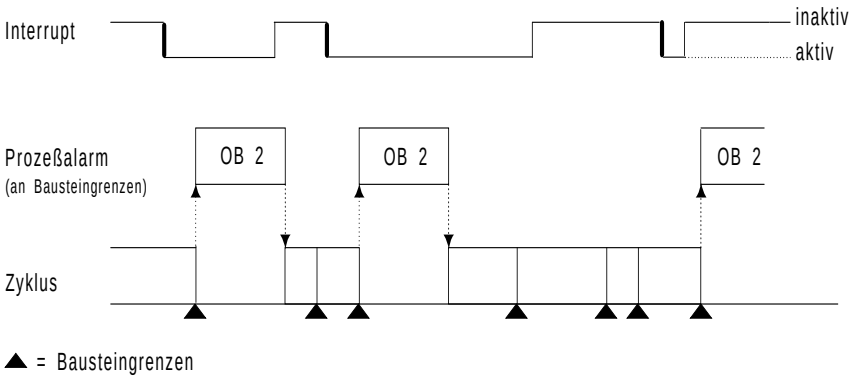


Bild 4-8 flankengetriggerte Prozeßalarmsignale

Sperren der prozeßalarmgesteuerten Bearbeitung

Ein alarmgesteuertes Programm wird an einer Bausteingrenze oder einer STEP-5-Befehlsgrenze in das zyklische Programm eingeschoben. Diese Unterbrechung kann sich ungünstig auswirken, wenn ein zyklischer Programmteil in einer bestimmten Zeit bearbeitet werden muß (um z.B. eine bestimmte Reaktionszeit zu erreichen) oder eine Befehlsfolge nicht unterbrochen werden darf (z. B. beim Lesen oder Schreiben von zusammengehörenden Werten).

Wenn ein Programmteil durch eine alarmgesteuerte Bearbeitung **nicht** unterbrochen werden darf, kommen folgende Programmiermöglichkeiten in Frage:

- Programmieren Sie diesen Programmteil so, daß er keinen Bausteinwechsel enthält und behalten Sie die Voreinstellung im DX 0 ("Prozeßalarme an Bausteingrenzen") bei. Programmteile, die keinen Bausteinwechsel enthalten, können dann auch nicht unterbrochen werden.

- Programmieren Sie den STEP-5-Befehl ’AS’ (Prozeßalarme sperren). Mit dem Befehl ’AF’ (Prozeßalarme freigeben) geben Sie die Alarmbearbeitung wieder frei. Zwischen diesen beiden Befehlen wird keine alarmgesteuerte Programmbearbeitung durchgeführt, der dazwischen stehende Programmteil kann durch auftretende Prozeßalarme nicht unterbrochen werden.
  ’AS’ und ’AF’ sind nur in Funktionsbausteinen möglich (Ergänzender Operationsvorrat)!

- Verwenden Sie die Sonderfunktionen OB 120 und OB 122, mit denen Sie die Bearbeitung von auftretenden Prozeßalarmen für einen bestimmten Programmteil sperren oder verzögern können.

4.5.5 Verschachtelte alarm- und zeitgesteuerte Programmbearbeitung

Prioritierung von alarm- und zeitgesteuerter Programmbearbeitung

Wenn während einer zeitgesteuerten Programmbearbeitung ein Prozeßlarm auftritt, wird das Programm an der nächsten Unterbrechungsstelle (Baustein- oder Befehlsgrenze) unterbrochen und der Prozeßalarm bearbeitet. Danach wird die zeitgesteuerte Programmbearbeitung zu Ende geführt.

Wenn während der alarmgesteuerten Programmbearbeitung ein Weckalarm auftritt, wird zuerst die alarmgesteuerte Programmbearbeitung abgeschlossen. Erst dann wird die zeitgesteuerte Programmbearbeitung aufgenommen.

Wenn **gleichzeitig** ein Prozeßalarm und ein Weckalarm auftreten, dann wird an der nächsten Unterbrechungsstelle zuerst der Prozeßalarm bearbeitet. Erst wenn dieser abgearbeitet ist, wird der noch anstehende Weckalarm bearbeitet.

Bild 4-9 zeigt schematisch, wie die Unterbrechung der Programmbearbeitung an Bausteingrenzen durch zeitgesteuerte und programmgesteuerte Alarmbearbeitung erfolgt.

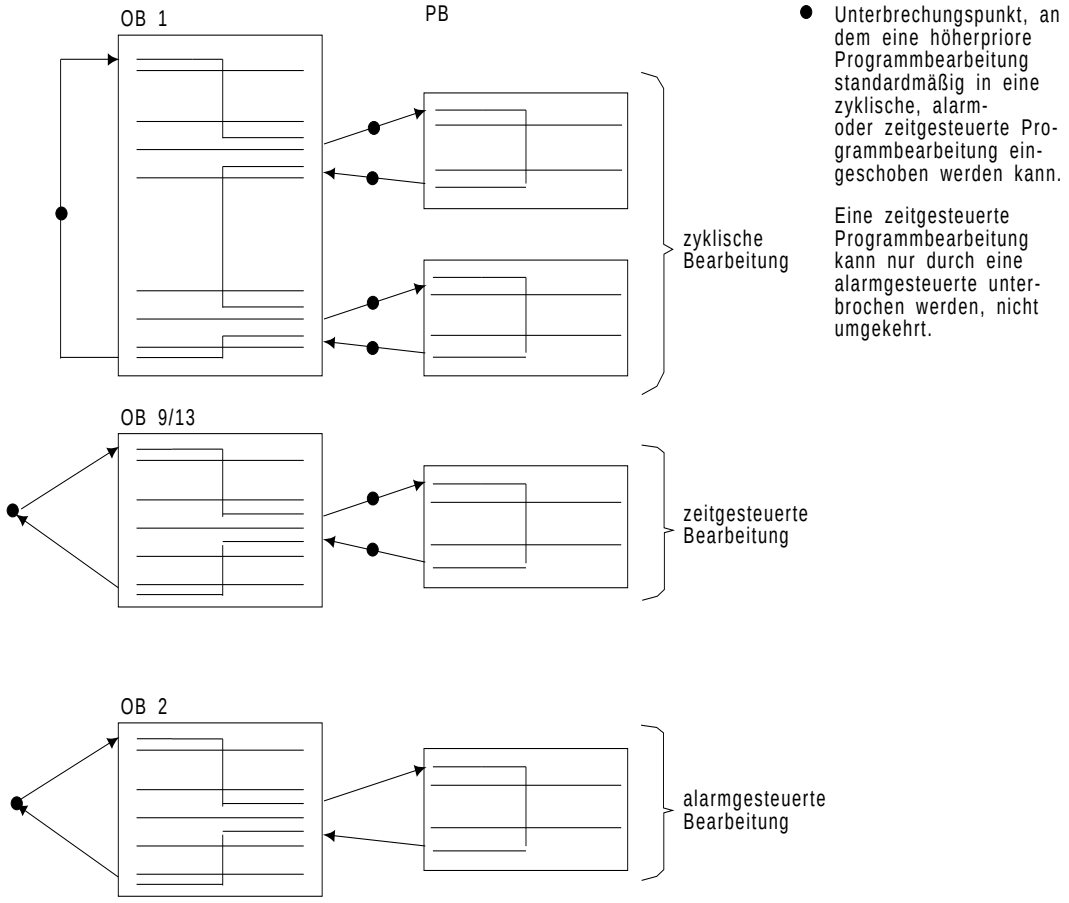


Bild 4-9 Unterbrechungsgesteuerte Programmbearbeitung an Bausteingrenzen

Reaktionszeit

Die Reaktionszeit auf eine Weckalarmanforderung entspricht der Bearbeitungszeit eines Bausteins bzw. eines STEP-5-Befehls (je nach gewählter Voreinstellung). Wenn jedoch zum Zeitpunkt der Unterbrechung der zyklischen Programmbearbeitung noch Prozeßalarme anstehen, wird das zeitgesteuerte Programm erst dann bearbeitet, nachdem alle anstehenden Prozeßalarme vollständig abgearbeitet sind.

Die maximale Reaktionszeit zwischen dem Auftreten und der Bearbeitung eines Weckalarms wächst in diesem Fall um die Bearbeitungszeit der Prozeßalarme. Wollen Sie für einen bestimmten Weckalarm-OB xy einen Weckfehler weitgehend ausschließen, beachten Sie folgende Regel:

$A + B + C < D$ wobei

A = Summe der Bearbeitungszeiten aller höherprioren Programmbearbeitungsebenen (Prozeß-, Regler-, Weckalarm-OBs)

B = Bearbeitungszeit des Weckalarm-OB xy

C = Laufzeit des längsten Bausteins aller niederprioren Bearbeitungsebenen

D = Zeitraster des Weckalarm-OB xy

Hinweis
Wenn Sie Ihr Anwenderprogramm nicht nur zyklisch, sondern auch zeit- und/oder alarmgesteuert bearbeiten lassen, besteht die Gefahr, daß Merker überschrieben werden.
Dies kann z. B. geschehen, wenn diese sowohl im zyklischen als auch im eingeschachtelten Programm als Zwischenmerker verwendet werden, und die zyklische Programmbearbeitung durch ein eingeschachteltes zeit- oder alarmgesteuertes Programm unterbrochen wird.

Aus diesem Grund müssen Sie zu Beginn einer zeit- oder alarmgesteuerten Programmbearbeitung die Signalzustände der (doppelt belegten) Merker in einen Datenbaustein "retten" und am Ende der unterbrechenden Bearbeitung wieder zurückschreiben.

Zu diesem Zweck stehen Ihnen vier Sonderfunktions-Organisationsbausteine zur Verfügung: OB 190 und OB 192 "Merker in Datenbaustein übertragen" und OB 191 und 193 "Datenblöcke in Merkerbereich übertragen" (siehe dort).

Um eine Doppelbelegung von Merkern zu vermeiden, können Sie für die meisten Anwendungsfälle auch S-Merker verwenden. Spezielle "Rettungsmaßnahmen" für Merker sind dann nicht erforderlich (S-Merker stehen in genügender Anzahl zur Verfügung).

Unterbrechungs- und Fehlerbehandlung

5

Inhalt von Kapitel 5

5 Unterbrechungs- und Fehlerbehandlung

In diesem Kapitel erfahren Sie, wie Sie Fehler beim Planen und Programmieren Ihrer STEP-5-Programme vermeiden können.
Sie werden informiert, welche Hilfen Ihnen das Systemprogramm zur Fehlerdiagnose und evtl. Fehlerreaktion zur Verfügung stellt und in welchen Bausteinen Sie Reaktionen auf bestimmte Fehler programmieren können.

5.1 Häufige Fehler im Anwenderprogramm

Das Systemprogramm kann fehlerhaftes Arbeiten der CPU, Fehler in der Systemprogrammbearbeitung oder Auswirkungen einer fehlerhaften Programmierung durch den Anwender feststellen.

Die folgende Liste enthält eine Aufzählung von Fehlern, die bei der Inbetriebnahme des Anwenderprogramms am häufigsten auftreten, die Sie jedoch schon beim Erstellen Ihres Programms leicht vermeiden können.
Beachten Sie aus diesem Grund beim Entwurf, bei der Implementierung und bei der Inbetriebnahme Ihres STEP-5-Programms bitte folgende Punkte:

- Bei der Angabe von Byte-Adressen für Ein- und Ausgänge müssen entsprechend adressierte Baugruppen im Zentralgerät oder Erweiterungsgerät stecken.

- Achten Sie darauf, daß alle Operanden mit den korrekten Parametern versorgt werden.

- Vorsicht beim nachträglichen Ändern von Funktionsbausteinen. Kontrollieren Sie, ob die FB/FX mit den richtigen Operanden parametriert sind und ob alle Aktualoperanden angegeben sind.

- Ausgänge, Merker, Zeiten und Zähler sollten nicht an mehreren Stellen im Programm mit entgegengesetzt wirkenden Operationen bearbeitet werden.

- Zeiten sollten nur einmal im Zyklus abgefragt werden (z. B. U T1).

- Sorgen Sie dafür, daß alle im Programm aufgerufenen Datenbausteine vorhanden und ausreichend lang sind.

- Überprüfen Sie, ob alle aufgerufenen Bausteine auch tatsächlich im Speicher vorhanden sind.

- Schmiermerker sollten von alarm- und zeitgesteuerten Programmen gerettet und am Schluß der Bearbeitung wieder geladen werden, wenn diese von anderen Bausteinen (z. B. Standard-FB) benötigt werden.

5.2 Fehlerinformationen

Wenn im Anlauf oder bei der zyklischen Bearbeitung des Anwenderprogramms ein Fehler auftritt, haben Sie verschiedene "Informationsquellen" zur Verfügung, um diesem Fehler auf die Spur zu kommen. Dies sind:

- LEDs auf der Frontplatte der CPU
- Unterbrechungsstack USTACK und Steuerbits
- Systemdatum BS 3, BS 4 und BS 80
- Fehlerkennungen in AKKU 1 und AKKU 2
- Bausteinstack BSTACK

In den nachfolgenden Abschnitten können Sie nachlesen, welche Hilfsmittel es zur Auswertung dieser Informationsquellen gibt und wie Sie die Fehlerinformation für die Analyse einer Störung verwenden können.

LEDs auf der Frontplatte der CPU

Orientieren Sie sich im Falle eines unerwünschten Stoppzustandes an den Leuchtdioden. Sie können Ihnen Hinweise auf die Fehlerursache geben:

LED-Anzeige	Bedeutung
STOP-LED zeigt Dauerlicht	Die unterschiedlichen Anzeigen der STOP-LED deuten auf bestimmte Unterbrechungs- und Fehlerursachen hin. Beachten Sie hierzu die Ausführungen im Abschnitt 4.1.
STOP-LED blinkt langsam	
STOP-LED blinkt schnell	
ADF-LED zeigt Dauerlicht	Adressierfehler
QVZ-LED zeigt Dauerlicht	Quittungsverzug
ZYK-LED zeigt Dauerlicht	Zykluszeitfehler

PG-Online-Funktion AUSGABE USTACK

Die PG-Online-Funktion AUSGABE USTACK gibt Ihnen Auskunft über die Zustände CPU-interner Steuerbits und den Inhalt des Unterbrechungsstacks (USTACK).

In den **USTACK** trägt das Systemprogramm beim Übergang in den Stoppzustand alle Informationen ein, die es für einen Wiederanlauf benötigt. Diese Informationen sind:

- Registerinhalte,
- Akkuinhalte,
- STEP-Adreßzähler SAZ

  und

- Ergebnisanzeigen.

Diese Einträge sind bei der Fehlerdiagnose eine wertvolle Hilfe.

Vor der Ausgabe des USTACK-**Inhalts** werden zunächst die Zustände der **Steuerbits** angezeigt. Diese markieren den aktuellen Betriebszustand, bestimmte Eigenschaften der CPU und des Anwenderprogramms und geben zusätzliche Hinweise auf die Fehlerursache.

Sie können die Funktion "Ausgabe USTACK" in den Betriebszuständen STOP, ANLAUF und RUN benutzen; allerdings erhalten Sie im ANLAUF und im RUN nur Auskunft über die Steuerbits und nicht über den Inhalt des USTACKS.

Die Bedeutung der Steuerbits und der Aufbau des Unterbrechungsstacks sind in Abschnitt 5.3 ausführlich beschrieben.

Systemdaten BS 3 und BS 4

Wenn Ihre CPU schon beim **ANLAUF** infolge eines Fehlers zurück in den Stoppzustand geht, so wird in den Systemdatenwörtern BS 3 und BS 4 die Fehlerursache genauer definiert (siehe Abschnitt 5.5). In diesem Fall handelt es sich um Fehler, auf die das Systemprogramm beim Aufbau der Adreßlisten im DB 0 oder bei der Auswertung des DB 1, DB 2, DX 0 oder DX 2 stößt.

Die beiden Datenwörter sind unter folgenden absoluten Speicheradressen hinterlegt:

Systemdatenwort BS 3: KH = EA03

Systemdatenwort BS 4: KH = EA04

Über die Fehlerkennung im Systemdatenwort BS 3 können Sie feststellen, **was** für ein Fehler aufgetreten ist.
Das Systemdatenwort BS 4 gibt Ihnen Auskunft darüber, **wo** der Fehler aufgetreten ist.

Die Fehlerkennungen sind im Datenformat KH eingetragen.

Auswertung von Systemdatenwort BS 3 und BS 4 mit dem Programmiergerät

Mit der Online-Funktion AUSKUNFT ADRESSE (KH = EA03 bzw. EA04) können Sie den Inhalt der beiden Systemdatenwörter direkt auslesen und so die Fehlerursache ermitteln.

Systemdatum BS 80

Wenn vom Systemprogramm ein schwerer Systemfehler festgestellt wird, so setzt es das Steuerbit INF im Unterbrechungsstack (siehe Abschnitt 5.3) und hinterlegt im Systemdatenwort BS 80 eine zusätzliche Fehlerkennung im Datenformat KH.

Das Systemdatemwort BS 80 hat die absolute Speicheradressse KH = EA50. Sie können es in derselben Weise auslesen wie die Systemdaten BS 3 und BS 4.

Fehlerkennungen in AKKU 1 und AKKU 2

Treten bei der STEP-5-Programmbearbeitung im **ANLAUF** oder im **ZYKLUS** Fehler auf, für die als Anwenderschnittstelle ein bestimmter Organisationsbaustein vorhanden ist, so hinterlegt das Systemprogramm automatisch beim Aufruf des jeweiligen Organisationsbausteins in den Akkumulatoren AKKU 1 und AKKU 2 zusätzliche Fehlerinformationen, die die Fehlerursache näher erläutern (siehe Abschnitt 5.6).

Über die Fehlerkennung im AKKU 1 können Sie feststellen, **was** für ein Fehler aufgetreten ist.

Eine Fehlerkennung im AKKU 2 (falls eingetragen) gibt Ihnen Auskunft darüber, **wo** der Fehler aufgetreten ist.

Fehlerkennungen sind im Datenformat KH eingetragen.

Auswertung von AKKU 1 und AKKU 2 mit dem Programmiergerät

Mit der Online-Funktion AUSGABE USTACK können Sie den Inhalt der beiden Akkumulatoren direkt aus dem USTACK lesen und so die genaue Fehlerursache ermitteln.

Auswertung von AKKU 1 und AKKU 2 mit STEP 5

Da die Fehlerkennungen automatisch beim Aufruf eines Fehler-Organisationsbausteins im AKKU 1 und 2 abgelegt werden, können Sie diese Kennungen bei der Programmierung Ihres Fehler-OBs berücksichtigen.
Es ist somit möglich, in einem Organisationsbaustein unterschiedliche Reaktionen auf verschiedene Fehler vorzusehen, in Abhängigkeit von der dort übergebenen Fehlerkennung.

Online-Funktion AUSGABE BSTACK

Die PG-Online-Funktion AUSGABE BSTACK gibt Ihnen Auskunft im STOP über den Inhalt des Bausteinstacks (BSTACK – siehe Abschnitt 3.2 "Bausteinschachtelung").

Im BSTACK sind, ausgehend vom OB 1 bzw. FB 0, alle Bausteine aufgeführt, die **nacheinander** bis zum Übergang in den Stoppzustand aufgerufen und noch nicht zu Ende bearbeitet worden sind. Da der BSTACK von unten gefüllt wird, steht in der obersten Zeile der BSTACK-Ausgabe derjenige Baustein, der als **letzter** bearbeitet wurde und in dem der Fehler aufgetreten ist.

BSTACK-Informationen

Bei der Auswertung der **obersten** Zeile erhalten Sie folgende Informationen:

Information	Bedeutung
BAUST.-NR.	Bausteinart und -nummer des Bausteins, der den fehlerhaften Baustein aufgerufen hat
BAUST.-ADR.	absolute Anfangsadresse dieses Bausteins im Anwenderspeicher
RÜCKSPR.-ADR.	Absolutadresse der nächsten zu bearbeitenden STEP-5-Operation dieses Bausteins im Anwenderspeicher
REL.-ADR	Relativadresse (= Differenz "RÜCKSPR.-ADR. - BAUST.-ADR.") der nächsten zu bearbeitenden Operation in diesem Baustein (Relativadressen können vom PG in Betriebsart "Eingabesperre"/Schlüsselschalter und mit S5-DOS ab Stufe IV über Funktionstaste "Adressen" angezeigt werden.)
DB-NR.	Nummer des in diesem Baustein aufgeschlagenen Datenbausteins
DB-ADR	absolute Anfangsadresse dieses Datenbausteins (Adresse des Datenwortes DW 0) im Programmspeicher

Beispiel

AUSGABE BSTACK auswerten:

BAUST.-NR.	BAUST.-ADR.	RUECKSPR.-ADR.	REL.-ADR	DB-NR.	DB-ADR.
OB 23	0063	0064	0001	13	0078
FB 5	006A	0072	0008	13	0078
FB 6	008A	0091	0007	100	0098
OB 1	009D	009E	0001		

Im diesem Beispiel ist der Stoppzustand im OB 23 bei der Bearbeitung derjenigen STEP-5-Anweisung aufgetreten, die im Speicher unter der Absolutadresse "0064 - 1 = 0063" abgelegt ist.

Der OB 23 (QVZ-Fehler-OB) ist im FB 5 an der relativen Adresse "0008 - 1 = 0007" aufgerufen worden.

Im FB 6 ist der Datenbaustein DB 100 aufgeschlagen worden. Beim Übergang der CPU in den Stoppzustand war Datenbaustein DB 13 gültig.

Der Datenbaustein DB 13 wurde im FB 5 aufgeschlagen.

5.3 Steuerbits und Unterbrechungsstack

Über die Online-Funktionen AG-INFO und AUSGABE USTACK können Sie mit dem Programmiergerät sowohl Betriebszustand, Eigenschaften der CPU und des Anwenderprogramms als auch eventuelle Fehler- und Unterbrechungsursachen analysieren.

Hinweis
Die Ausgabe der **Steuerbits** erhalten Sie in **jedem** Betriebszustand, die Ausgabe des **USTACK-Inhalts** nur im **STOP**.

- **Steuerbits** geben den aktuellen bzw. vorausgegangenen Betriebszustand und die Störungsursache an.
  Sind mehrere Fehler aufgetreten, werden in den Steuerbits **alle aufgetretenen** Fehler dargestellt.

- Im **USTACK** wird die jeweilige Unterbrechungsstelle (Adressen) mit
  den dort aktuellen Anzeigen und Akkuinhalten sowie die Störungsursache angegeben.
  Sind mehrere Unterbrechungen aufgetreten, so wird ein **mehrstufiger** Unterbrechungsstack aufgebaut:

  Tiefe 01 = letzte Unterbrechungsursache,

  Tiefe 02 = vorletzte Unterbrechungsursache usw.

  Bei USTACK-Überlauf (nach mehr als 13 Einträgen) erfolgt ein sofortiger Übergang in den Stoppzustand. Anschließend ist NETZ-AUS/NETZ EIN und ein NEUSTART erforderlich

Die Bedeutung der einzelnen Abkürzungen in den Steuerbits und im Unterbrechungsstack wird nachfolgend erläutert.

Hinweis
Der Text am Bildschirm Ihres Programmiergerätes ist abhängig von der benutzten PG-Software. Daher kann er von der hier abgedruckten Darstellung abweichen. Trotzdem ist die Beschreibung der Bildschirminformation in dieser Programmieranleitung gültig!

5.3.1 Steuerbits

Bei der PG-Ausgabe des USTACKs werden auf der 1. Bildschirmseite die Zustände der Steuerbits angezeigt (siehe Bild 5-1).

S T E U E R B I T S

>>STP<<	STP-6	FE-STP	BARBEND	PG-STP	STP-SCH	STP-BEF	MP-STP
>>ANL<<	ANL-6	NEUSTA X	M W A	A W A	ANL-2	NEUZU X	MWA-ZUL X
>>RUN<< X	RUN-6	EINPROZ X	BARB	OB1GEL	FB0GEL X	OBPROZA	OBWECKA
32KWRAM	16KWRAM	8KWRAM X	EPROM	KM-AUS	KM-EIN	DIG-EIN X	DIG-AUS X
URGELOE	URL-IA	STP-VER	ANL-ABB	UA-PG	UA-SYS	UA-PRFE	UA-SCH
DX0-FE	FE-22	MOF-FE	RAM-FE	DB0-FE	DB1-FE	DB2-FE	KOR-FE
N A U	P E U	B A U	STUE-FE	Z Y K	Q V Z	A D F	WECK-FE
B C F	FE-6	FE-5	FE-4	FE-3	L Z F	REG-FE	DOPP-FE

Bild 5-1 Beispiel für die 1. Bildschirmseite "AUSGABE USTACK": Steuerbits

Bei der PG-Ausgabe des USTACKs werden auf der 1. Bildschirmseite die Zustände der Steuerbits angezeigt.

Die Steuerbits (»STP«, »ANL« und »RUN«) und die in den ersten drei Zeilen der 1. Bildschirmseite folgenden Bits markieren den aktuellen bzw. vorausgegangenen Betriebszustand der CPU und geben Auskunft über bestimmte Eigenschaften der CPU und des STEP-5-Anwenderprogramms.

Die Steuerbits lassen sich in allen Betriebszuständen ausgeben. So können Sie sich z.B. jederzeit vergewissern, ob der Organisationsbaustein OB 2 geladen und somit eine alarmgesteuerte Programmbearbeitung möglich ist.

In den nachfolgenden Tabellen finden Sie, welche Aussage die einzelnen Bits haben.

In den nachfolgenden Tabellen finden Sie, welche Aussage die einzelnen Bits haben.

Tabelle 5-1 Bedeutung der Steuerbits in der Zeile >>STP<<

Zeile >>STP<< (STEUERBITS)	
Steuerbit	**Bedeutung**
>>STP<<	CPU ist im Betriebszustand STOP
STP-6	nicht belegt
FE-STP	Fehler-Stop: Stoppzustand nach NAU (Netzausfall), PEU (Peripherie unklar), BAU (Batterie unklar), STUEB (BSTACK-Überlauf), STUEU (USTACK-Überlauf), DOPP (Doppelfehler) oder CPU-Fehler.
BARBEND	Bearbeitungskontrolle-Ende: Stoppzustand nach Online-Funktion BEARBEITUNGSKONTROLLE ENDE (NEUSTART erforderlich). Wird nicht gesetzt, wenn die Funktion BEARBEITUNGSKONTROLLE ENDE im Stoppzustand der CPU durchgeführt wurde.
PG-STP	PG-STOP: Stoppzustand durch Befehl vom PG
STP-SCH	STOP-Schalter: Stoppzustand durch Betriebsartenschalter in Stellung STOP
STP-BEF	Stopp-Befehl: - Stoppzustand nach Bearbeitung der STEP-5-Operation 'STP', - Stoppzustand nach Stoppbefehl vom Systemprogramm, wenn Fehler-Organisationsbaustein nicht programmiert ist.
MP-STP	Mehrprozessor-STOP: - Wahlschalter am KOR in Stellung STOP, - STOP einer anderen CPU im Mehrprozessorbetrieb.

Tabelle 5-2 Bedeutung der Steuerbits in der Zeile >>ANL<<

Zeile >>ANL<< (STEUERBITS)	
Steuerbit	**Bedeutung**
>>ANL<<	CPU ist im Betriebszustand ANLAUF
ANL-6 + M W A	MANUELLER NEUSTART MIT GEDÄCHTNIS
ANL-6 + AWA	AUTOMATISCHER NEUSTART MIT GEDÄCHTNIS
NEUSTA	MANUELLER NEUSTART ist angefordert (STOP) oder wurde als letzter ANLAUF durchgeführt (ANLAUF/RUN).
M W A	MANUELLER WIEDERANLAUF ist angefordert (STOP) oder wurde als letzter ANLAUF durchgeführt (ANLAUF/RUN).
A W A	AUTOMATISCHER WIEDERANLAUF nach Netzspannungsausfall ist angefordert (STOP) oder wurde als letzter ANLAUF durchgeführt (ANLAUF/RUN).
M W A + A W A	AUTOMATISCHER NEUSTART wurde angefordert (STOP) oder als letzte Anlaufart wurde AUTOMATISCHER NEUSTART durchgeführt (ANLAUF/RUN)
ANL-2	Doppelfunktion: - wird gesetzt nach Aufruf von BEARBEITUNGSKONTROLLE-ENDE (wird im Unterschied zu BARBEND in der ersten Maskenzeile auch dann gesetzt, wenn BEARBEITUNGSKONTROLLE-ENDE im STOP aufgerufen wird; verhindert WIEDERANLAUF), - wird gesetzt nach "Komprimieren im STOP"; verhindert WIEDERANLAUF.
NEUZU	NEUSTART zulässig (STOP) oder beim letzten ANLAUF **war** NEUSTART zulässig (ANLAUF/RUN).
MWA-ZUL	MANUELLER WIEDERANLAUF zulässig (STOP) oder beim letzten ANLAUF **war** NEUSTART zulässig (ANLAUF/RUN).

Tabelle 5-3 Bedeutung der Steuerbits in der Zeile >>RUN<<

Zeile >>RUN<< (STEUERBITS)	
Steuerbit	**Bedeutung**
>>RUN<<	CPU ist im Zustand RUN (zykl. Bearbeitung aktiv)
RUN-6	nicht belegt
EINPROZ	Einzelprozessorbetrieb
BARB	Online-Funktion BEARBEITUNGSKONTROLLE ist aktiv.
OB1GEL	Organisationsbaustein OB 1 ist in den Anwenderspeicher geladen worden. Die zyklische Programmbearbeitung wird durch den OB 1 bestimmt.
FB0GEL	Funktionsbaustein FB 0 ist in den Anwenderspeicher geladen worden. Die zyklische Programmbearbeitung wird durch den FB 0 bestimmt, wenn kein OB 1 geladen ist. Wenn FB 0 und OB 1 geladen sind, gilt der OB 1 für die zyklische Programmbearbeitung.
OBPROZA	Prozeßalarm-Organisationsbaustein OB 2 geladen, d. h. prozeßalarm- gesteuerte Programmbearbeitung möglich
OBWECKA	Weckalarm-Organisationsbaustein geladen, d. h. zeitgesteuerte Programmbearbeitung möglich

Tabelle 5-4 Bedeutung der Steuerbits in den Zeilen 4 und 5

Zeilen 4 und 5 (STEUERBITS)	
Steuerbit	**Bedeutung**
32KWRAM	Anwenderspeichermodul ist ein RAM mit 32 x 2^{10} Wörtern
16KWRAM	Anwenderspeichermodul ist ein RAM mit 16 x 2^{10} Wörtern
8KWRAM	Anwenderspeichermodul ist ein RAM mit 8 x 2^{10} Wörtern
EPROM	Anwenderspeichermodul ist ein EPROM
KM-AUS	Adreßliste für Koppelmerkerausgänge aus DB 1 vorhanden.
KM-EIN	Adreßliste für Koppelmerkereingänge aus DB 1 vorhanden

Zeilen 4 und 5 (STEUERBITS)	
Steuerbit	**Bedeutung**
Fortsetzung der Tabelle 5-4:	
DIG-EIN	Adreßliste für digitale Eingänge vorhanden
DIG-AUS	Adreßliste für digitale Ausgänge vorhanden
URGELOE	CPU wurde urgelöscht (NEUSTART erforderlich).
URL-IA	CPU wird momentan urgelöscht.
STP-VER	CPU hat Stoppzustand des AG verursacht.
ANL-ABB	ANLAUF wurde abgebrochen (NEUSTART erforderlich).
UA-PG	PG hat URLÖSCHEN angefordert.
UA-SYS	Systemprogramm hat URLÖSCHEN angefordert (kein ANLAUF möglich); URLÖSCHEN muß durchgeführt werden.
UA-PRFE	Urlöschanforderung wegen CPU-Fehler
UA-SCH	Urlöschvoranforderung durch Schalterbedienung: URLÖSCHEN durchführen bzw. Wahl einer Anlaufart, wenn angefordertes URLÖSCHEN nicht durchgeführt werden soll.

Die Steuerbits der Zeilen 6 bis 8 (siehe nachfolgende Tabelle) markieren Fehler, die in den Betriebszuständen ANLAUF (z.B. beim ersten NEUSTART) und RUN (z. B. bei der zeitgesteuerten Programmbearbeitung) auftreten können.
Sind mehrere Fehler aufgetreten, werden in den letzten drei Zeilen der Steuerbits **alle** bisher aufgetretenen (und noch nicht bearbeiteten!) Unterbrechungsursachen angezeigt. **Beachten Sie hierzu das Systemdatum BS 2:** Es enthält das UAMK (Unterbrechungsanzeigen-Sammelwort, 16 bit), in dem ebenfalls alle aufgetretenen und noch nicht bearbeiteten Fehler eingetragen sind (Abschnitt 8.3.5).

Tabelle 5-5 Bedeutung der Steuerbits in den Zeilen 6 bis 8

Zeilen 6 bis 8 (STEUERBITS)	
Steuerbit	**Bedeutung**
DX0-FE	Parametrierfehler im DX 0 oder DX 2
FE-22	nicht belegt
MOD-FE	Inhalt des Anwendermoduls ist fehlerhaft (URLÖSCHEN erforderlich).
RAM-FE	Inhalt des Systemprogramm-RAMs oder des DB-RAMs ist fehlerhaft (URLÖSCHEN erforderlich).
DB0-FE	Aufbau der Baustein-Adreßlisten im DB0 ist fehlerhaft.
DB1-FE	Aufbau der Adreßlisten im DB 1 für Prozeß-abbild-Aktualisierung ist fehlerhaft: - DB 1 bei gestecktem Koordinator oder bei Mehrprozessorbetrieb nicht programmiert, - Aufbau oder Inhalt der DB 1 ist fehlerhaft.
DB2-FE	Fehler bei der Auswertung des Parametrierungs-Datenbausteins DB 2 der Reglerstruktur R64
KOR-FE	Fehler beim Datenaustausch mit dem Koordinator
N A U	Netzspannungsausfall im Zentralgerät
P E U	Peripherie unklar = Spannungsausfall im Erweiterungsgerät
B A U	Batterie ist fehlerhaft = Ausfall der Pufferbatterie im Zentralgerät.
STUE-FE	Unterbrechungs- oder Bausteinstack übergelaufen (Schachtelungstiefe zu groß; NEUSTART erforderlich)
Z Y K	Zyklusüberwachungszeit überschritten
Q V Z	Quittungsverzug beim Datenaustausch mit Peripherie
A D F	Adressierfehler bei Eingängen oder Ausgängen: Fehler hervorgerufen durch Zugriff auf das Prozeß-abbild, wobei Peripheriebaugruppen angesprochen wurden, die beim letzten NEUSTART nicht gesteckt, defekt oder nicht im DB 1 angegeben waren.
WECK-FE	Weckfehler: Vor oder während der Bearbeitung eines bestimmten Weckalarm-OBs ist ein weiterer Weckalarm für diesen OB ausgelöst worden

Zeilen 6 bis 8 (STEUERBITS)	
Steuerbit	**Bedeutung**
Fortsetzung der Tabelle 5-5:	
B C F	Befehlscodefehler: - Substitutionsfehler: Bearbeiteter STEP-5-Befehl ist nicht substituierbar, - Operationscodefehler: Bearbeiteter STEP-5-Befehl ist falsch, - Parameterfehler: Parameter des bearbeiteten STEP-5-Befehls ist falsch.
FE-6	nicht belegt
FE-5	Hinweis auf schweren Systemfehler, Zusatzinformation in BS 80
FE-4	Power-down-Fehler: Bearbeitung eines vorausgegangenen Netzausfalls (NAU) durch das Systemprogramm ist fehlerhaft abgelaufen; der WIEDERANLAUF ist deswegen gesperrt.
FE-3	Schnittstellenfehler (SSF)
L Z F	Laufzeitfehler: - Aufgerufener Baustein nicht geladen, - Lade-/Transferfehler bei Datenbausteinen, - Sonstige Laufzeitfehler.
REG-FE	Fehler bei der Bearbeitung der Reglerstruktur R64 im ZYKLUS
DOPP-FE	Doppelfehler: Eine noch aktive Fehlerprogrammbearbeitungsebene (ADF, BCF, LZF, QVZ, REG, ZYK) wird ein zweites Mal aktiviert (NEUSTART erforderlich).

5.3.2 USTACK-Inhalt

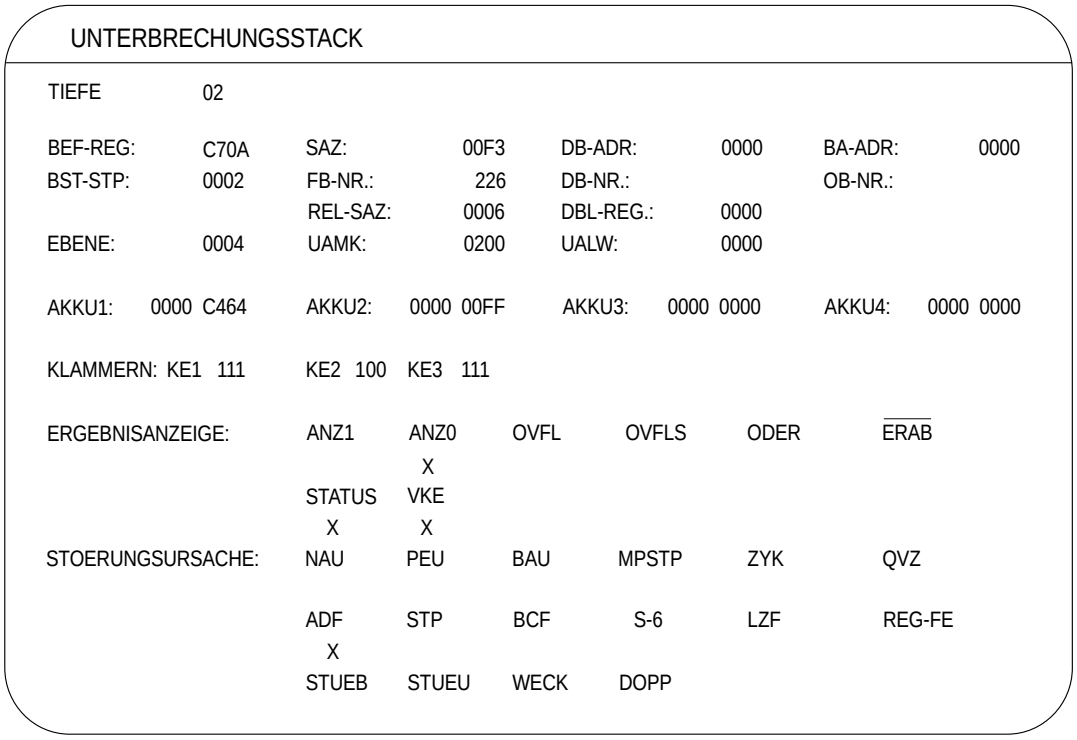
Befindet sich die CPU im Stoppzustand, so erscheint nach Ausgabe der Steuerbits und Betätigen der Übernahmetaste auf dem Bildschirm des PGs der Inhalt des USTACKS. Das Systemprogramm trägt beim Übergang in den Stoppzustand alle Informationen in den USTACK ein, die es für einen Wiederanlauf benötigt.

Aus den USTACK-Informationen können Sie bei der Fehlerdiagnose entnehmen, was für ein Fehler aufgetreten ist und an welcher Stelle im Programm er sich ausgewirkt hat.

Hat **ein einziger** Fehler den Stoppzustand verursacht, so wird nur **eine** Ebene mit USTACK-Informationen ausgegeben. Sind **mehrere** Fehler augetreten, so werden **entsprechend viele** Ebenen mit USTACK-Informationen angezeigt (TIEFE 01, TIEFE 02 usw.). In allen Ebenen ist unter STOERUNGSURSACHE ein einziger Fehler angekreuzt.

Bei mehreren Fehlern ist in der Ebene TIEFE 01 der Fehler markiert, der unmittelbar vor dem Übergang in den Stoppzustand festgestellt wurde.

Ein Beispiel für die PG-Ausgabe des USTACK-Inhalts sehen Sie in Bild 5-2.



UNTERBRECHUNGSSTACK

TIEFE	02								
BEF-REG:	C70A	SAZ:	00F3	DB-ADR:	0000	BA-ADR:	0000		
BST-STP:	0002	FB-NR.:	226	DB-NR.:		OB-NR.:			
		REL-SAZ:	0006	DBL-REG.:	0000				
EBENE:	0004	UAMK:	0200	UALW:	0000				
AKKU1:	0000 C464	AKKU2:	0000 00FF	AKKU3:	0000 0000	AKKU4:	0000 0000		
KLAMMERN:	KE1 111	KE2	100	KE3	111				
ERGEBNISANZEIGE:		ANZ1	ANZ0	OVFL	OVFLS	ODER	$\overline{\text{ERAB}}$		
			X						
		STATUS	VKE						
		X	X						
STOERUNGSURSACHE:		NAU	PEU	BAU	MPSTP	ZYK	QVZ		
		ADF	STP	BCF	S-6	LZF	REG-FE		
		X							
		STUEB	STUEU	WECK	DOPP				

Bild 5-2 Beispiel für eine Bildschirmseite "AUSGABE USTACK": Inhalt

Erläuterung der USTACK-Anzeigen

TIEFE

Stufe der Informationsebene des USTACK-Inhalts bei Fehlerschachtelung:

TIEFE 01 = zuletzt aufgetretene Störungsursache,
TIEFE 02 = vorletzte aufgetretene Störungsurache
.....
TIEFE13 = ... (maximale Tiefe)

Angaben über die Fehlerstelle

Die folgenden Tabelle enthält Angaben (USTACK-Kennungen) über die Fehlerstelle, mit denen im Anwenderprogramm die Anweisung gefunden werden kann, bei deren Bearbeitung die CPU in STOP gegangen ist.

Tabelle 5-6 Bedeutung der USTACK-Kennungen zur Fehlerstelle

Angaben zur Fehlerstelle	
USTACK-Kennung	**Bedeutung**
BEF-REG	Befehlsregister: Es enthält den Machinencode (1. Wort des zuletzt bearbeiteten Befehls einer unterbrochenen Programmbearbeitungsebene siehe Operationsliste, Auflistung des Maschinencodes.
BST-STP	Bausteinstack-Pointer: Er enthält die Anzahl der im Bausteinstack (BSTACK) eingetragenen Elemente zum Zeitpunkt der Unterbrechung dieser Programmbearbeitungsebene.
EBENE Z	Gibt die Ebene der Programmbearbeitung an, die unterbrochen worden ist: Z: 0002: NEUSTART 0004: ZYKLUS 0006: WECKAL./5 s (OB 18) 0008: WECKAL./2 s (OB 17) 000A: WECKAL./ s (OB 16) 000C: WECKAL./500 ms (OB 15) 000E: WECKAL./200 ms (OB 14) 0010: WECKAL./100 ms (OB 13) 0012: WECKAL./50 ms (OB 12) 0014: WECKAL./20 ms (OB 11) 0016: WECKAL./10 ms (OB 10) 0018: ZEITAUFTRAG

Angaben zur Fehlerstelle	
USTACK-Kennung	**Bedeutung**
Fortsetzung 1 der Tabelle 5-6:	
EBENE Z (Forts.)	Z: 001A: nicht belegt 001C: REGLERALARM 001E: nicht belegt 0020: VERZÖGERUNGSALARM 0022: nicht belegt 0024: PROZESSALARM 0026: nicht belegt 0028: MANUELLER NEUSTART MIT GEDÄCHTNIS 002A: AUTOMATISCHER NEU-START MIT GEDÄCHTNIS 002C: Übergang in den Stoppzustand bei STOP im Mehrprozessor-betrieb, Stoppschalter oder PG-STOP 002E: Schnittstellenfehler 0030: Weckfehler 0032: Reglerfehler 0034: Zyklusfehler 0036: nicht belegt 0038: Befehlscodefehler 003A: Laufzeitfehler 003C: Adressierfehler 003E: Quittungsverzug 0040: nicht belegt 0042: nicht belegt 0044: MANUELLER WIEDERANLAUF 0046: AUTOMATISCHER WIEDERANLAUF
SAZ	STEP-Adreßzähler: - Enthält die **Absolut**adresse des zuletzt bearbeiteten Befehls einer unterbrochenen Programmbearbeitungsebene im **Programmspeicher**. - Bei Fehler zeigt der SAZ genau auf den fehlerverursachenden Befehl! - Vor Ausführung des ersten Befehls einer Bearbeitungsebene steht der SAZ auf ’0’.
...NR.	Bausteinart und -nummer des zuletzt bearbeiteten Bausteins

Angaben zur Fehlerstelle	
USTACK-Kennung	**Bedeutung**
Fortsetzung 2 der Tabelle 5-6:	
REL-SAZ	Relativer STEP-Adreßzähler: Enthält die **Relativ**adresse (bezogen auf die Bausteinanfangsadresse) des zuletzt bearbeiteten Befehls im zuletzt bearbeiteten Baustein (Relativadressen können vom PG in Betriebsart "Eingabesperre"/ Schlüsselschalter oder mit S5-DOS ab Stufe IV über Funktionstaste angezeigt werden oder bei der Ausgabe des Bausteins auf den Drucker).
UAMK	Unterbrechungsanzeigen-Sammelwort: Im UAMK sind alle bisher aufgetretenen und noch nicht zu Ende bearbeiteten Unterbrechungsursachen angezeigt (siehe Abschnitt 8.3.5).
UALW	Unterbrechungsanzeigen-Löschwort (siehe Abschnitt 8.3.5)
DB-ADR	absolute Anfangsadresse (DW 0) des zuletzt aufgeschlagenen Datenbausteins im Programmspeicher (= 0000, wenn kein Datenbaustein aufgeschlagen wurde)
DB-NR	Nummer des zuletzt aufgeschlagenen Datenbausteins
DBL-REG	Länge des zuletzt aufgeschlagenen Datenbausteins
BA-ADR	Absolutadresse im Programmspeicher für den nächsten zu bearbeitenden Befehl im **zuletzt** aufrufenden Baustein
...NR.	Bausteinart und -nummer des zuletzt aufrufenden Bausteins
AKKU1 bis AKKU4	Inhalt der Rechenregister zum Unterbrechungszeitpunkt: In bestimmten Fehlerfällen werden vom Systemprogramm in AKKU 1 und AKKU 2 Fehlerkennungen hinterlegt, die Unterbrechungsursache näher erläutern.

Angaben zur Fehlerstelle	
USTACK-Kennung	**Bedeutung**
Fortsetzung 3 der Tabelle 5-6:	
KLAMMERN	Anzahl der Klammerebenen: "KEx abc" mit x = 1 bis 7 Ebenen, a = 'OR' (Oder, siehe Bitanzeigen), b =' VKE' (Verknüpfungsergebnis, siehe Bitanzeigen), c = 1: 'U(', c = 0: 'O('.

ERGEBNISANZEIGE siehe Abschnitt 3.5

STOERUNGSURSACHE Die folgenden Abkürzungen (USTACK-Kennungen) stellen die wichtigsten Störungsursachen dar.
Es sind nur diejenigen Unterbrechungsursachen angekreuzt, die in der gerade angezeigten Programmbearbeitungsebene (siehe EBENE!) aufgetreten sind.
Bei den Angaben der Störungsursachen handelt es sich um die Wiedergabe des Unterbrechungsanzeigensammelwortes (UAMK, 16 bit; siehe Abschnitt 8.3.5). Teilweise sind die Angaben hier mit denen der Steuerbits identisch.

Tabelle 5-7 USTACK-Kennungen STOERUNGSURSACHE

STOERUNGSURSACHE	
USTACK-Kennung	**Bedeutung (aufgerufener Fehler-OB)**
NAU	Netzspannungsausfall im Zentralgerät
PEU	Peripherie unklar = Spannungsausfall im Erweiterungsgerät
BAU	Batterie unklar = Ausfallder Puferbatterie (Zentralgerät)
MPSTP	Mehrprozessor-STOP: - Wahlschalter am KOR in Stellung STOP oder - STOP einer anderen CPU im Mehrprozessorbetrieb

STOERUNGSURSACHE	
USTACK-Kennung	**Bedeutung (aufgerufener Fehler-OB)**
Fortsetzung 1 der Tabelle 5-7:	
ZYK	Zyklusüberwachungszeit überschritten
QVZ	Quittungsverzug beim Datenaustausch mit der Peripherie)
ADF	Adressierfehler bei digitalen Eingängen und Ausgängen mit Prozeßabbild
STP	- Stoppzustand durch Stoppschalter in Stellung STOP - Stoppzustand durch Befehl vom PG - Stoppzustand nach Bearbeitung der STEP-5-Operation 'STP' - Stoppzustand nach Stoppbefehl vom System-programm, wenn Fehler-Organisationsbaustein nicht programmiert ist
BCF	Befehlscodefehler: Fehler, die während der Befehlsdekodierung erkannt werden: - Substitutionsfehler: Bearbeiteter STEP-5-Befehl ist nicht substituierbar - Operationscodefehler: Bearbeiteter STEP-5-Befehl ist falsch Parameterfehler: Parameter des bearbeiteten STEP-5-Befehls ist unzulässig
S-6	Schnittstellenfehler
LZF	Laufzeitfehler: Fehler, die während der Befehlsausführung erkannt werden: - Aufgerufener Baustein nicht geladen - Lade-Transferfehler bei Datenbausteinen - Sonstige Laufzeitfehler
REG-FE	Fehler bei der Bearbeitung der Reglerstruktur R64 im ZYKLUS
STUEB	Der Bausteinstack ist übergelaufen: Die Schachtelungstiefe ist zu groß; erforderliche Maßnahme: NEUSTART
STUEU	Der Unterbrechungsstack ist übergelaufen: Die Schachtelungstiefe ist zu groß; erforderliche Maßnahmen: NEUSTART

STOERUNGSURSACHE	
USTACK-Kennung	**Bedeutung (aufgerufener Fehler-OB)**
Fortsetzung 2 der Tabelle 5-7:	
WECK	Weckfehler: Vor oder während der Bearbeitung eines bestimmten Weckalarm-OBs ist ein weiterer Weckalarm für diesen OB ausgelöst worden
DOPP	Doppelfehler: Eine noch aktive Fehlerprogrammbearbeitungs-ebene (ADF, BCF, LZF, QVZ, REG-FE, ZYK) wird ein zweites Mal aktiviert (NEUSTART erforderlich)

5.3.3 Beispiele zur Fehlerdiagnose über USTACK

Beispiel 1:

Bild 5-3 zeigt Ihnen den Aufbau des USTACK in Zusammenhang mit den aufgetretenen Unterbrechungen.

- Die Programmbearbeitungsebene ZYKLUS (**OB 1**) wird unterbrochen durch das Auftreten eines Interrupts.
- Daraufhin wird die Programmbearbeitungsebene WECKALARM aktiviert und der **OB 13** bearbeitet.
- Durch das Auftreten eines Prozeßalarms wird die Ebene WECKALARM verlassen, die Ebene PROZESSALARM aktiviert und der **OB 2** bearbeitet.
- Ein falscher Adressierbefehl führt dazu, daß die Ebene ADF aktiviert und dort der OB 25 bearbeitet wird. In seinem Fehlerbehandlungsprogramm hat der Anwender einen Stoppbefehl (STP) programmiert: Die CPU bricht die Programmbearbeitung ab.

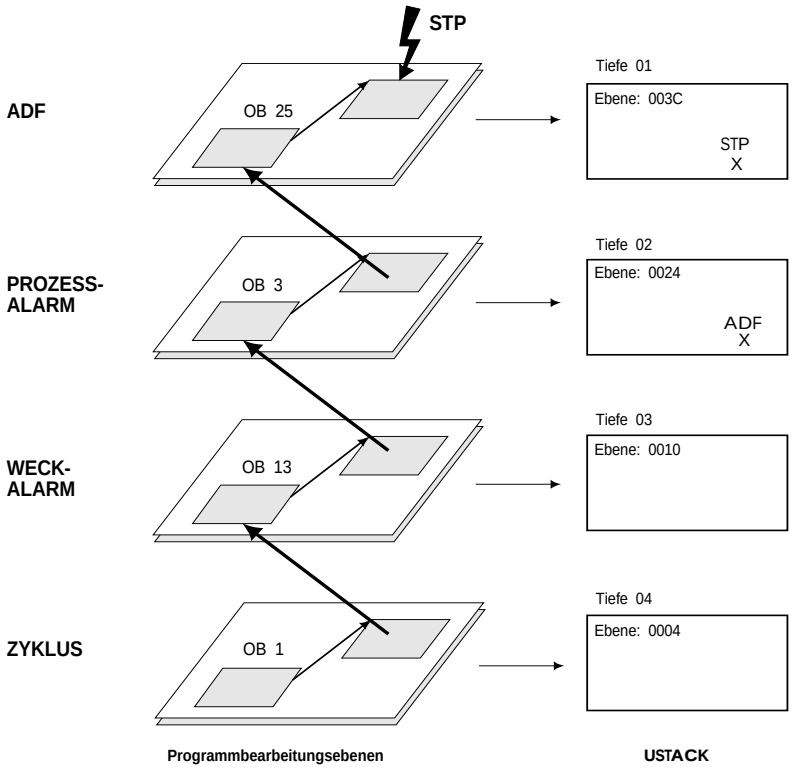


Bild 5-3 Beispiel 1 zur Auswertung des USTACKs

Vor dem endgültigen Übergang in den Stoppzustand sind insgesamt 4 verschiedene Programmbearbeitungsebenen unterbrochen worden. Wenn Sie sich nun am PG den USTACK ausgeben lassen, bekommen Sie entsprechend einen **vierstufigen** USTACK, zuoberst der USTACK mit der Tiefe 01, in dem die Kennung der **zuletzt** unterbrochenen Programmbearbeitungsebene (= ADF) vermerkt ist. Sie können den USTACK nun "hinunterschalten" bis zum USTACK mit der Tiefe 04, der die Programmbearbeitungsebene ZYKLUS repräsentiert, die **als erste** unterbrochen wurde.

Beispiel 2:

In diesem Beispiel erkennt die CPU bei der Ausführung der Operation U Ex.y im OB 1 einen Adressierfehler. Dies führt zur Bearbeitung des OB 25. Augrund einer STP-Operation im PB 5 geht die CPU in den Stoppzustand (siehe Bild 5-4).

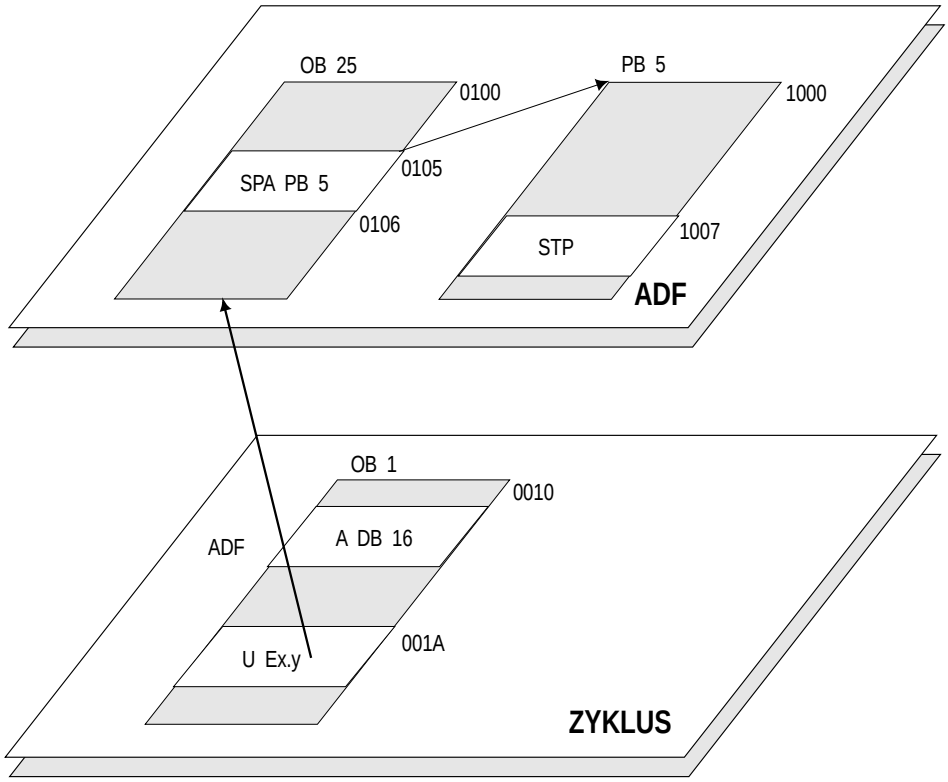


Bild 5-4 Beispiel 2 zur Auswertung des USTACKs

Fortsetzung auf der nächsten Seite

Fortsetzung 1 von Beispiel 2:

Zwei unterbrochene Programmbearbeitungsebenen führen zum Aufbau eines zweistufigen USTACKs (siehe Bilder 5-5 und 5-6):

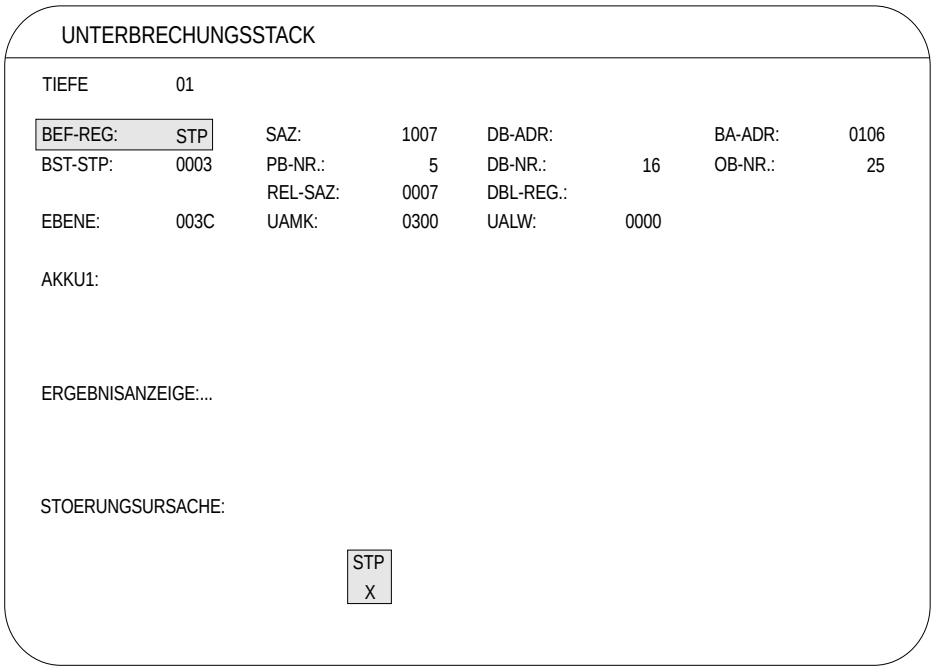


Bild 5-5 Beispiel 2 zur Auswertung des USTACKs: 1. USTACK-Ebene

Fortsetzung auf der nächsten Seite

Fortsetzung 2 von Beispiel 2:

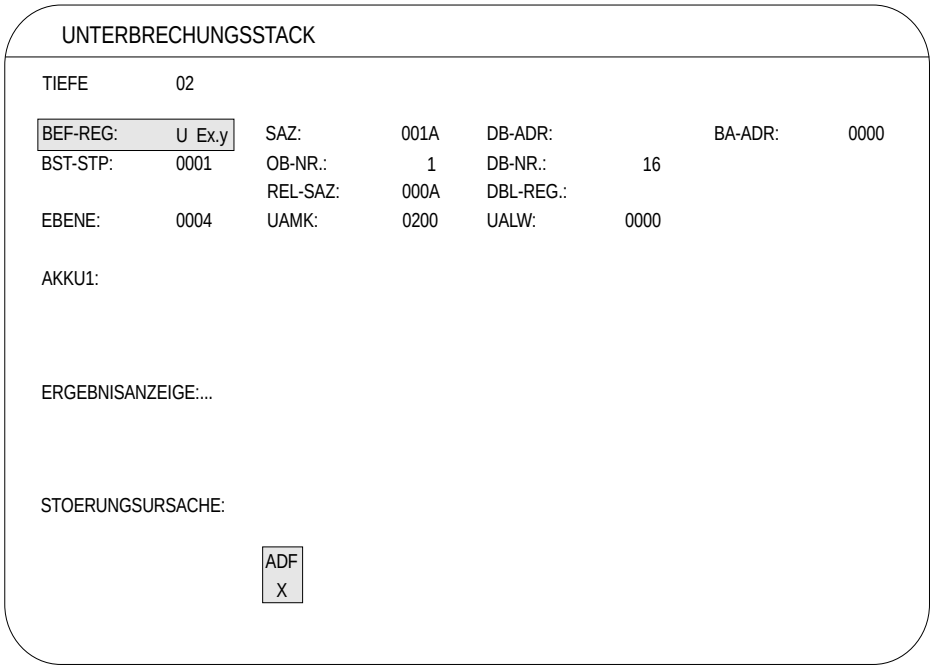


Bild 5-6 Beispiel 2 zur Auswertung des USTACKs: 2. USTACK-Ebene

5.4 Fehlerbehandlung über Organisationsbausteine

Wenn das Systemprogramm einen bestimmten Fehler erkannt hat, ruft es den für diesen Fall vorgesehenen Organisationsbaustein auf. Durch entsprechende Programmierung dieses Organisationsbausteins können Sie nun das weitere Verhalten der CPU festlegen.
Abhängig davon, wie Sie den Organisationsbaustein programmieren, können Sie

- die normale Programmbearbeitung fortsetzen lassen,
- die CPU in STOP bringen

  und/oder

- ein spezielles "Fehlerprogramm" bearbeiten lassen.

Für die folgenden Fehlerursachen sind Organisationsbausteine vorhanden:

Tabelle 5-8 Bei Fehlern aufgerufene Organisationsbausteine

Fehlerursache	Aufruf von	Reaktion bei nicht geladenem OB [1)]
Aufruf eines nicht geladenen Bausteins (LZF)	OB 19	STOP
Quittungsverzug im Anwenderprogramm bei Zugriff auf Peripheriebaugruppen (QVZ)	OB 23	keine
Quittungsverzug beim Aktualisieren des Prozeßabbildes und bei Koppelmerkerübertragung (QVZ)	OB 24	keine
Adressierfehler (ADF)	OB 25	STOP
Zykluszeitüberschreitung (ZYK)	OB 26	STOP
Substitutionsfehler (BCF)	OB 27	STOP
Betriebsartenschalter auf STOP, PG-Funktion AG-STOP, STOP vom S5-Bus (Mehrprozessorbetrieb)	OB 28	STOP
Operationscodefehler (BCF)	OB 29	STOP
Parameterfehler (BCF)	OB 30	STOP
Sonstige Laufzeitfehler (LZF)	OB 31	STOP
Lade-/Transferfehler bei Datenbausteinen (TRAF)	OB 32	STOP
Weckfehler (WECK-FE)	OB 33	STOP

Fehlerursache	Aufruf von	Reaktion bei nicht geladenem OB [1]
Fortsetzung der Tabelle 5-8:		
Fehler bei der Bearbeitung der Reglerstruktur R64 (REG-FE)	OB 34	STOP
Kommunikationsfehler auf der zweiten seriellen Schnittstelle (FE-3)	OB 35	keine

[1] mit DX-0-Voreinstellung.

Reaktion bei nicht geladenem Organisationsbaustein

Die Reaktion bei **nicht gelademem** Organisationsbaustein ist fehlerabhängig:

Keine Unterbrechung der zyklischen Programmbearbeitung

Tritt ein Quittungverzug auf und OB 23, OB 24 oder OB 35 sind nicht geladen, so wird die zyklische Programmbearbeitung **nicht** unterbrochen. Es erfolgt keine Reaktion der CPU.

Soll die CPU bei QVZ in den Stoppzustand übergehen, so muß der Organisationsbaustein eine Stopp-Anweisung enthalten und mit BE abgeschlossen oder der DX 0 entsprechend parametrietriert werden.

Programm für STOP:

```
:
:
:STP
:BE
```

Stoppzustand

In allen übrigen Fehlerfällen geht die CPU sofort in den Stoppzustand, wenn die zugehörigen Organisationsbausteine vom Anwender nicht programmiert worden sind.

Soll in Ausnahmefällen (z. B. während der Inbetriebsetzung) der eine oder andere Fehler die zyklische Programmbearbeitung nicht unterbrechen, so genügt eine Bausteinende-Anweisung im jeweiligen Organisationsbaustein oder eine entsprechende Parametrierung des DX 0.

Programm für Betrieb ohne Unterbrechung:

```
:
:
:BE
```

Hinweis
Eine Ausnahme bildet der Organisationsbaustein OB 28: Hier erfolgt immer ein Übergang in den Stoppzustand, unabhängig davon, ob der OB 28 geladen ist und wie er programmiert ist.

Wenn Sie den betreffenden Organisationsbaustein nicht programmieren wollen, haben Sie die Möglichkeit, durch entsprechende Parametrierung des **Datenbausteins DX 0** den Übergang der CPU in den Stoppzustand zu verhindern.

Unterbrechungen bei der Bearbeitung der Fehler-Organisationsbausteine

Nachdem das Systemprogramm den betreffenden Organisationsbaustein aufgerufen hat, wird das darin enthaltene Anwenderprogramm bearbeitet.
Tritt während der Bearbeitung eines Organisationsbausteins erneut ein Fehler auf, wird wie in der zyklischen Programmbearbeitung das Programm an der nächsten Befehlsgrenze unterbrochen und der entsprechend andere Organisationsbaustein aufgerufen.

Die Organisationsbausteine werden in der Reihenfolge bearbeitet, in der sie aufgerufen werden. Wieviele Fehler-Organisationsbausteine ineinandergeschachtelt werden können, ist abhängig von

- **der Art der aufgetretenen Fehler:**

  Es können keine Organisationsbausteine ineinandergeschachtelt werden, die derselben Programmbearbeitungsebene angehören. (Zur Zuordnung der Fehler-OB zu den Programmbearbeitungsebenen siehe folgendes Kapitel.)

  Bei der Bearbeitung des OB 27 (Programmbearbeitungsebene BCF) kann beispielsweise ein OB 32 (Programmbearbeitungsebene LZF), nicht jedoch ein OB 29 oder OB 30 (ebenfalls BCF) eingeschachtelt werden.

  Bei Doppelaufruf einer Programmbearbeitungsebene geht die CPU unmittelbar in STOP.

- **der Anzahl der zu diesem Zeitpunkt aktivierten Programmbearbeitungsebenen:**

  Für jede aktivierte Programmbearbeitungsebene benötigt das Systemprogramm bei Unterbrechungen besonderen Speicherplatz zum Anlegen des USTACKs. Reicht dieser Speicherplatz nicht mehr aus, so kommt es zu einem USTACK-Überlauf.

  Bei USTACK-Überlauf geht die CPU unmittelbar in STOP.

- **der Anzahl der zu diesem Zeitpunkt aufgerufenen Bausteine:**

  Bei BSTACK-Überlauf geht die CPU unmittelbar in STOP.

5.5 Fehler im ANLAUF

Bei der Initialisierung und im Anlauf auftretende Störungs- und Fehlerursachen können dazu führen, daß das Anlaufprogramm abgebrochen wird und die CPU in den Stoppzustand übergeht.
Im Anlaufprogramm (Organisationsbausteine OB 20, 21 und 22) auftretende Unterbrechungsursachen werden wie im ZYKLUS behandelt.

Ausnahme: Bei einem STOP im Anlauf wird **kein** Organisationsbaustein OB 28 aufgerufen.

Unterbrechungs- und Fehlerursachen

Auf die in der folgenden Tabelle genannten Unterbrechungs- und Fehlerursache gibt es **keine Reaktionsmöglichkeit** durch eine Anwenderschnittstelle (Fehler-OB).

Tabelle 5-9 Fehler- und Unterbrechungsursachen im ANLAUF

Steuerbit oder Kennung im USTACK	Erläuterung
STP	Stoppbefehl vom Systemprogramm (bei FE-STP) oder im Anwenderprogramm
BAU	Ausfall der Pufferbatterie am Zentralgerät
NAU	Ausfall der Versorgungsspannung im Zentralgerät
PEU	Ausfall der Versorgungsspannung in einem Erweiterungsgerät
STUEU	Stacküberlauf beim Unterbrechungsstack (USTACK)
STUEB	Stacküberlauf beim Bausteinstack (BSTACK)
DOPP-FE	Doppelaufruf einer Fehlerprogrammbearbeitungsebene
RAM-FE	Fehler bei der Initialisierung: Inhalt des Betriebssystem-RAMs oder des DB-RAMs defekt
MOD-FE	Fehler bei der Initialisierung: Inhalt des Anwendermoduls (RAM- oder EPROM-Modul) nicht korrekt
DB0-FE [1)]	Fehler beim Aufbau der Bausteinadreßliste (DB 0)
DB1-FE [1)]	Fehler bei der Auswertung des DB 1 zum Aufbau der Adreßliste für die Prozeßabbildaktualisierung

Steuerbit oder Kennung im USTACK	Erläuterung
Fortsetzung der Tabelle 5-9:	
DB2-FE [1]	Fehler bei der Auswertung des DB 2 der Reglerstruktur R64
DX0-FE [1]	Fehler bei der Auswertung des Datenbausteins DX 0 oder Fehler bei der Auswertung des Datenbausteins DX 2

[1] Diese Fehler werden nachfolgend näher erläutert.

5.5.1 DB0-FE (DB-0-Fehler)

Fehler beim Aufbau der Bausteinadreßliste (Datenbaustein DB 0)

Der DB 0 wird vom Systemprogramm nach NETZ EIN aufgebaut. Bei einem DB-0-Fehler finden Sie in den Systemdatenwörtern BS 3 und BS 4 Fehlerkennungen, die den aufgetretenen Fehler näher definieren.

Tabelle 5-10 Kennungen für DB-0-Fehler

Fehlerkennung BS 3	BS 4	Erläuterung
8001H	yyyyH	Falsche Bausteinlänge yyyy =Adresse des Bausteins mit falscher Länge
8002H	yyyyH	Berechnete Endadresse des Bausteins im Speicher falsch yyyy =Bausteinadresse
8003H	yyyyH	Ungültige Bausteinkennung yyyy =Adresse des Bausteins mit falscher Kennung
8004H	yyyyH	Zu große Organisationsbausteinnummer (erlaubt: OB 1 bis OB 39) yyyy =Adresse des Bausteins mit falscher Nummer
8005H	yyyyH	Datenbausteinnummer 0 (erlaubt: DB 1 bis DB 255) yyyy =Adresse des Bausteins mit falscher Nummer

5.5.2 DB1-FE (DB-1-Fehler)

Fehler bei der Auswertung des DB 1 zum Aufbau der Adreßliste für die Prozeßabbildaktualisierung

- fehlender DB 1 im Mehrprozessorbetrieb

  oder

- fehlerhafte DB1-Adreßliste bei NEUSTART.

Hinweis
Im Mehrprozessorbetrieb wird bei **allen** Anlaufarten geprüft, ob der DB 1 vorhanden ist. Die **Auswertung** der DB-1-Parameter erfolgt jedoch nur bei NEUSTART!

Tabelle 5-11 Kennungen für DB-1-Fehler

Fehlerkennung BS 3	BS 4	Erläuterung
0410H	yyyyH	Unzulässige Kennung: - Kopfkennung fehlt oder fehlerhaft (korrekt KC MASK01) - Kennung unzulässig (zulässig KH DE00, DA00, CE00, CA00, BB00) - Endekennung fehlt oder fehlerhaft (korrekt KH EEEE) yyyy = unzulässige Kennung
0411H	yyyyH	"Digitale Eingänge", Anzahl Adressen unzulässig (zulässig 0...128) yyyy = unzulässige Anzahl Adressen
0412H	yyyyH	"Digitale Ausgänge", Anzahl Adressen unzulässig (zulässig 0...128) yyyy = unzulässige Anzahl Adressen
0413H	yyyyH	"Koppelmerker-Eingänge", Anzahl Adressen unzulässig (zulässig 0...256) yyyy = unzulässige Anzahl Adressen
0414H	yyyyH	"Koppelmerker-Ausgänge", Anzahl Adressen unzulässig (zulässig 0...256) yyyy = unzulässige Anzahl Adressen
0415H	yyyyH	Ungültige Anzahl Zeitzellen (erlaubt: 256) yyyy = unzulässige Anzahl Zeitzellen
0419H	yyyyH	Quittungsverzug bei digitalen Eingängen yyyy = Adresse des nicht quittierten Eingangsbytes
041AH	yyyyH	Quittungsverzug bei digitalen Ausgängen yyyy = Adresse des nicht quittierten Ausgangsmerkerbytes

Fehlerkennung BS 3	BS 4	Erläuterung
Fortsetzung der Tabelle 5-11:		
041BH	yyyyH	Quittungsverzug bei Koppelmerker-Eingang yyyy = Adresse des nicht quittierten Koppelmerkerbytes
041CH	yyyyH	Quittungsverzug bei Koppelmerker-Ausgang yyyy = Adresse des nicht quittierten Koppelmerkerbytes

5.5.3 DB2-FE (DB-2-Fehler)

Fehler bei der Auswertung des Parametrierungs-Datenbaustein DB 2 der Reglerstruktur R64 (Reglerinitialisierung).

Bei einem DB-2-Fehler finden Sie in den Systemdatenwörtern BS 3 und BS 4 Fehlerkennungen, die den aufgetretenen Fehler näher definieren.

Tabelle 5-12 Kennungen für DB-2-Fehler

Fehlerkennung BS 3	BS 4	Erläuterung
0421H	DByyH	Datenbaustein ist nicht geladen yy = Nummer des nicht geladenen Datenbausteins
0422H	FByyH	Funktionsbaustein nicht geladen yy = Nummer des nicht geladenen Funktionsbausteins
0423H	FByyH	Funktionsbaustein nicht erkannt yy = Nummer des nicht erkannten Funktionsbausteins
0424H	FByyH	Funktionsbaustein mit falscher PG-Software geladen yy = Nummer des Funktionsbausteins
0425H	DByyH	Falsche Regler-Datenbaustein-Länge yy = Nummer des Datenbausteins
0426H	–	Für das Verschieben der Regler-DBs vom Anwender-EPROM in das DB-RAM ist der Speicherplatz im DB-RAM nicht ausreichend

5.5.4 DX0-FE (DX-0- oder DX-2-Fehler)

Hinweis
DX-0-Fehler und DX-2-Fehler haben ein gemeinsames Steuerbit (DX0-FE) in der Steuerbit-Maske.

Fehler bei der Auswertung des Datenbausteins DX 0

Bei einem DX-0-Fehler finden Sie in den Systemdatenwörtern BS 3 und BS 4 Fehlerkennungen, die den aufgetretenen Fehler näher definieren.

Tabelle 5-13 Kennungen für DX-0-Fehler

Fehlerkennung BS 3	BS 4	Erläuterung
0431H	yyyyH	Unzulässige Kennung: - Kopfkennung fehlt oder fehlerhaft (korrekt KC MASKX0) - Blockkennung unzulässig - Endekennung fehlt oder fehlerhaft (korrekt KH EEEE) yyyy = unzulässige Kennung
0432H	yyyyH	Unzulässiger Parameter yyyy = unzulässiger Parameter
0434H	yyyyH	Nicht erlaubte Anzahl Zeitzellen (erlaubt: 0...256) yyyy = falsche Anzahl Zeitzellen
0435H	yyyyH	Unerlaubte Zyklusüberwachungszeit (erlaubt: 1 ms bis 13000 ms) yyyy = falsche Zeitgröße

Fehler bei der Auswertung des Datenbausteins DX 2

Parametrierung der 2. seriellen Schnittstelle:
Der Datenbaustein DX 2 wird vom Systemprogramm bei NEU-START geprüft. Bei einem DX-2-Fehler finden Sie in den Systemdatenwörtern BS 3 und BS 4 Fehlerkennungen, die den aufgetretenen Fehler näher definieren.

Tabelle 5-14 Kennungen für DX-2-Fehler

Fehlerkennung BS 3	BS 4	Erläuterung
0451H	–	DX- 2-Länge (ohne Bausteinkopf) < 4 Wörter ist unzulässig
0452H	yyyyH	DX-2-Länge (ohne Bausteinkopf) ist für Kopplungstyp zu kurz yyyy = Länge DX 2

Fehlerkennung BS 3	BS 4	Erläuterung
Fortsetzung der Tabelle 5-14:		
0453H	yyyyH	Kopplungstyp unzulässig yyyy = Kopplungstyp
0454H	xx00H	Datenkennung für stat. Parametersatz ungültig (ungleich 44H, 58H) xx = Datenkennnung
0455H	xxyyH	Baustein für statischen Parametersatz unzulässig xx = Kennung / yy = DB-Nummer
0456H	xxyyH	Statischer Parametersatz nicht vorhanden xx = Kennung / yy = DB-Nummer
0457H y	yyyH	Statischer Parametersatz zu kurz yyyy = Nummer des nicht vorhandenen DW
0458H	xx00H	Datenkennung für dyn. Parametersatz ungültig (ungleich 44H, 58H, 00H) xx = Datenkennnung
0459H	xxyyH	Baustein für dyn. Parametersatz unzulässig xx = Kennung / yy = DB-Nummer
045AH	xx00H	Datenkennung für Sendefach/Auftragsfach ungültig (ungleich 44H, 58H, 00H) xx = Datenkennnung
045BH	xxyyH	Baustein für Sendefach/Auftragsfach unzulässig xx = Kennung / yy = DB-Nummer
045CH	xx00H	Datenkennung für Empfangsfach unglütig (ungleich 44H, 58H, 00H) xx = Datenkennnung
045DH	xxyyH	Baustein für Empfangsfach unzulässig xx = Kennung / yy = DB-Nummer
045EH	xx00H	Datenkennung für Koordinierungsbytes ungültig (ungleich 44H, 58H, 4DH) xx = Kennung
045FH	xxyyH	Baustein für Koordinierungsbytes unzulässig xx = Kennung / yy = DB-Nummer
0460H	xxyyH	Baustein für Koordinierungsbytes nicht vorhanden xx = Kennung / yy = DB-Nummer
0461H	yyyyH	Datenwort für Koordinierungsbytes nicht vorhanden yyyy = Nr. des nicht vorhandenen DW

5.6 Fehler im RUN und im ANLAUF

Im Betriebszustand RUN kann eine zyklische, eine zeit- oder alarmgesteuerte Programmbearbeitung oder eine Reglerbearbeitung an Befehlsgrenzen unterbrochen werden durch das Auftreten von Störungen wie z. B. Ausfall der Versorgungsspannung am Zentralgerät oder Überlauf des Bausteinstacks .

Bei der Initialisierung und im Betriebszustand ANLAUF auftretende Unterbrechungsursachen führen ebenfalls dazu, daß das Anlaufprogramm abgebrochen wird und die CPU in den Stoppzustand übergeht bzw. den für diesen Fehlerfall vorgesehenen Organisationsbaustein aufruft. Im Anlaufprogramm auftretende Unterbrechungsursachen werden wie im ZYKLUS behandelt.

Man unterscheidet zwischen Störungen, die die CPU direkt in den Betriebszustand STOP überführen (z. B. STUEU) und Störungen, bei deren Auftreten das Systemprogramm anstelle eines Übergangs in den Stoppzustand bestimmte Organisationsbausteine aufruft (z. B. ADF), die der Anwender programmieren kann.

Auf die in den folgenden beiden Tabellen genannten Unterbrechungs- und Fehlerursache gibt es **keine Reaktionsmöglichkeit** durch eine Anwenderschnittstelle (Fehler-OB).

Fehler, die direkt in den STOP führen

Bei diesen Fehlern wird ein USTACK aufgebaut, in dem die aufgetretene Störung angezeigt wird.

Tabelle 5-15 Fehler- und Unterbrechungsursachen im ANLAUF und RUN, die direkt in den STOP führen

Steuerbit oder Kennung im USTACK	Erläuterung
STP	STOP durch das Systemprogramm (bei Maschinenfehler), wenn ein Fehler-OB nicht geladen ist, oder Stoppbefehl im Anwenderprogramm
BAU	Ausfall der Pufferbatterie am Zentralgerät
NAU	Ausfall der Versorgungsspannung im Zentralgerät
PEU	Ausfall der Versorgungsspannung in einem Erweiterungsgerät
STUEU	Stacküberlauf beim Unterbrechungsstack bei zu großer Schachtelungstiefe (USTACK)
STUEB	Stacküberlauf beim Bausteinstack bei zu großer Schachtelungstiefe (BSTACK)
DOPP-FE	Doppelaufruf einer Fehlerprogrammbearbeitungsebene

Fehler, bei denen ein Fehler-OB aufgerufen wird

Tabelle 5-16 Fehler- und Unterbrechungsursachen im ANLAUF und RUN, bei denen ein Fehler-OB aufgerufen wird

Steuerbit oder Kennung im USTACK	Erläuterung	auf-gerufe-ner OB
BCF	Befehlscodefehler: - Substitutionsfehler - Operationscodefehler - Parameterfehler	 OB 27 OB 29 OB 30
LZF	Laufzeitfehler: - Aufruf eines nicht geladenen Bausteins - Transferfehler bei DBs - Sonstige Laufzeitfehler	 OB 19 OB 32 OB 31
ADF	Adressierfehler: - Bei Zugriff auf Prozeßabbild	 OB 25
QVZ	Quittungsverzug: - Im Anwenderprogramm bei Zugriff auf Peripheriebaugruppen - Bei der Prozeßabbildaktualisierung	 OB 23 OB 24
ZYK	Zyklusfehler: - Überschreitung der Zyklusüber-wachungszeit	 OB 26
WECK-FE	Weckfehler: - Fehler bei Bearbeitung eines Weckalarms	 OB 33
REG-FE	Reglerfehler: - Fehler bei Bearbeitung eines Regleralarms	 OB 34
ABBR	Abbruch: - (siehe Abschnitt 5.7.8)	 OB 28
S-6	Kommunikationsfehler: - Bei Datenverkehr über die zweite serielle Schnittstelle	 OB 35

Die folgenden Abschnitte beschreiben jede dieser Fehlerursachen genauer.

5.6.1 BCF (Befehlscodefehler)

Ein Befehlscodefehler tritt dadurch auf, daß die CPU einen STEP-5-Befehl des Anwenderprogramms nicht interpretieren oder ausführen kann. Alle zulässigen Befehlscodes sind in der Operationsliste aufgelistet.
Der Befehl, der den entsprechenden Befehlscodefehler verursacht, wird nicht ausgeführt. Falls der entsprechende BCF-Organisationsbaustein geladen ist, wird dieser aufgerufen, bearbeitet und anschließend mit dem nächsten Befehl im unterbrochenen Anwenderprogramm fortgefahren. Bei nicht geladenem BCF-OB geht die CPU in den Stoppzustand.

Es werden folgende Befehlscodefehler unterschieden, bei denen jeweils der genannte Fehler-OB augerufen wird:

Substitutionsfehler (OB 27)

Wenn in einem Funktionsbaustein eine Operation mit einem Formaloperanden ausgeführt werden soll, so ersetzt die CPU bei der Bearbeitung des Anwenderprogramms diesen Formaloperanden durch den im Aufruf des Funktionsbausteins stehenden Aktualoperanden.

Erkennt die CPU eine unzulässige Substitution, so unterbricht das Systemprogramm daraufhin die Bearbeitung des Anwenderprogramms und ruft den Organisationsbaustein **OB 27** auf, wenn dieser geladen ist.

AKKU 1 enthält zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-17 BCF – Substitutionsfehler

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1801H	–	Substitutionsfehler beim Befehl BBS
1802H	–	Substitutionsfehler bei den Befehlen DW, BMW
1803H	–	Substitutionsfehler bei den Befehlen B=, BI
1804H	–	Substitutionsfehler bei den Befehlen L=, T=
1805H	–	Substitutionsfehler bei den Befehlen U=, UN=, O=, ON=, ==, S= und RB=
1806H	–	Substitutionsfehler bei den Befehlen RD=, LC=, FR=, SAR=, SE=, SSV= und SVZ=

Operationscodefehler (OB 29)

Ein Operationscodefehler wird von der CPU bei der Bearbeitung eines STEP-5-Programms festgestellt, wenn ein Befehl programmiert worden ist, der nicht zum STEP-5-Befehlsumfang der CPU 928B gehört (z.B. können RU- und SU-Befehle mit dem PG programmiert, jedoch von den CPUs 928B, 928, 922 (R-Prozessor) und 921 (S-Prozessor) im AG 135U nicht interpretiert werden).

Beim Erkennen eines unzulässigen Operationscodes wird an dieser Stelle die Bearbeitung des Anwenderprogramms unterbrochen und der Organisationsbaustein **OB 29** aufgerufen, wenn er geladen ist.

Bei Aufruf des OB 29 stehen im AKKU 1 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-18 BCF – Operationscodefehler

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1811H	–	Befehl mit unzulässigem Opcode
1812H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 68H enthält
1813H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 78H enthält
1814H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 70H enthält
1815H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 60H enthält

Vorsicht
Ein Operationscodefehler sollte **nicht** quittiert werden: Die CPU erkennt nicht, ob es sich bei dem fehlerhaften Befehl um einen Einwort- oder Mehrwortbefehl handelt. Hat die CPU den OB 29 bearbeitet, versucht sie, das Programm mit dem nächsten Befehlswort fortzusetzen. Falls es sich dabei um das zweite Wort eines Mehrwortbefehls handelt, erkennt sie entweder einen weiteren Befehlscodefehler oder führt dieses Wort als gültigen Befehl aus, was zu beliebigen **Programmfehlern** führen kann.

Parameterfehler (OB 30)

Ein unzulässiger Parameter tritt auf, wenn ein Befehl mit einem Parameter, der für die entsprechende CPU unzulässig ist, programmiert worden ist (z. B. Aufruf eines reservierten Datenbausteins) oder wenn eine nicht vorhandene Sonderfunktion aufgerufen wird.

Wenn ein unzulässiger Parameter von der CPU erkannt wird, unterbricht das Systemprogramm die Bearbeitung des Anwenderprogramms und ruft den Organisationsbaustein **OB 30** auf, wenn dieser geladen ist.

Beim Aufruf des OB 30 stehen im AKKU 1 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-19 BCF – Parameterfehler

Fehlerkennung AKKU-1-L	Fehlerkennung AKKU-2-L	Erläuterung (unzulässiger Parameter bei ..)
1821H	–	A DB 0, 1, 2
182BH	–	SPA(B) OB 0
182CH	–	SPA(B) OB > 39: Sonderfunktion nicht vorhanden
182DH	–	AX DX 0, AX DX 1, AX DX 2
182EH	–	L MW/T MW/L PW/T PW/L QW/T QW/L DD/T DD/B MW 255
182FH	–	L EW/T EW/L AW/T AW 127
1830H	–	L MD/T MD 253, 254, 255
1831H	–	L ED/T ED/L AD/T AD 125, 126, 127
1832H	–	RLD/RRD/SVD/SLD 33-255
1833H	–	SLW/SRW/LIR/TIR 16-255
1834H	–	SES/SEF 32-255
1835H	–	U=/UN=/O=/ON=/S=/RB=/==/ RD=/FR=/SI=/SE=/SVZ=/SSV=/ SAR=/L=/LC=/LW=/T= 0, 127-255
1836H	–	B=/LD= 0, 126-255
1837H	–	U S/O S/S S/= S/UN S/ON S/R S Bytenummer > 1023
1838H	–	U S/O S/S S/= S/UN S/ON S/R S Bitnummer > 7
1839H	–	L SY/T SY Parameter>1023
183AH	–	L SW/T SW Parameter > 1022

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung (unzulässiger Parameter bei ..)
Fortsetzung der Tabelle 5-19:		
183BH	–	L SD/T SD Parameter >1020
183CH	–	E DB/EX DX Parameter 0, 1 oder 2 (DB bzw. DX 0, 1, 2 nicht erzeugbar)

5.6.2 LZF (Laufzeitfehler)

Ein Laufzeitfehler tritt dadurch auf, daß die CPU während der Bearbeitung eines STEP-5-Befehls einen Fehler erkennt.
Der Befehl, der den entsprechenden Laufzeitfehler verursacht, wird **nicht** ausgeführt (Ausnahme: Aufschlagen eines nicht vorhandenen Datenbausteins DB/DX). Ist ein LZF-Organisationsbaustein vorhanden, so wird dieser aufgerufen. Anschließend wird das unterbrochene Anwenderprogramm mit dem nächsten Befehl, der dem fehlerverursachenden Befehl folgt, fortgesetzt. Ist kein LZF-OB geladen, so geht die CPU in den STOP.

Es werden folgende Laufzeitfehler unterschieden bei denen jeweils der genannte Fehler-OB augerufen wird:

Aufruf eines nicht geladenen Bausteins (OB 19)

Wenn im Anwenderprogramm ein Baustein aufgerufen oder aufgeschlagen wird, der nicht vorhanden ist, erkennt das Systemprogramm einen Fehler. Dies gilt für alle Bausteinarten und sowohl für die bedingte als auch die unbedingte Aufruf-Anweisung.

Wenn der Aufruf oder das Aufschlagen eines nicht geladenen Bausteins erkannt wird, ruft das Systemprogramm den Organisationsbaustein **OB 19** auf, wenn er geladen ist. Im OB 19 können Sie das weitere Verhalten der CPU festlegen.

Falls ein OB 19 programmiert ist, wird dieser aufgerufen und anschließend die Bearbeitung des unterbrochenen STEP-5-Programms mit dem nächsten Befehl fortgesetzt außer, wenn der OB 19 mit einem Stoppbefehl programmiert ist. Ist hingegen der OB 19 nicht programmiert, so geht die CPU beim Aufruf oder Aufschlagen eines nicht geladenen Bausteins in den Stoppzustand.

Bei Aufruf des OB 19 stehen im AKKU 1 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-20 LZF – Aufruf eines nicht geladenen Bausteins

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A01H	–	Nicht geladener Datenbaustein bei A DB
1A02H	–	Nicht geladener Datenbaustein bei AX DX
1A03H	–	Nicht geladener Baustein bei SPA(B) FB, OB 1 bis OB 39, PB, SB
1A04H	–	Nicht geladener Baustein bei BA(B) FX
1A05H	–	Nicht geladener Datenbaustein bei OB 254 bzw. OB 255
1A06H	–	Nicht geladener Datenbaustein bei OB 182
1A07H	–	Nicht geladener Datenbaustein bei OB 150/OB 151/OB 153

Hinweis
Beim Versuch, einen nicht geladenen Datenbaustein aufzuschlagen, wird das DBA-Register (siehe Kapitel 9) beeinflußt. In diesem Fall muß vor Zugriffen auf DB/DX-Daten erneut ein geladener Datenbaustein aufgeschlagen werden.

Lade-/Transferfehler (OB 32)

Beim Transferieren von Daten in Datenbausteine (DB, DX) vergleicht die CPU die Länge des aufgeschlagenen DBs mit dem im Transfer-Befehl stehenden Operanden. Wird durch den angegebenen Parameter die Datenbausteinlänge überschritten, so wird die Transfer-Anweisung nicht ausgeführt, um ein irrtümliches Überschreiben von Daten im Speicher zu verhindern.

Ein Lade-/Transferfehler wird auch festgestellt, wenn ein einzelnes Bit innerhalb eines nicht vorhandenen Datenwortes abgefragt oder verändert werden soll.

Ein Lade-/Transferfehler wird ebenfalls erkannt, wenn ein Zugriff auf ein Datenwort stattfinden soll, bevor ein Datenbaustein aufgeschlagen ist (mit A DBn bzw. AX DXn).

Beim Erkennen eines Lade-/Transferfehlers ruft das Systemprogramm den Organisationsbaustein **OB 32** auf, wenn dieser geladen ist. Der Befehl, der den Transferfehler verursacht hat, wird nicht mehr bearbeitet. Bei Aufruf von OB 32 stehen im AKKU 1 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-21 LZF – Lade/Transferfehler (TRAF)

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A11H	–	Zugriff mit U/UN D, O/ON D, S/R D, =D auf ein nicht definiertes Datenwort
1A12H	–	Transferfehler bei TDR auf ein nicht definiertes Datenwort
1A13H	–	Transferfehler bei TDL auf ein nicht definiertes Datenwort
1A14H	–	Transferfehler bei TDW auf ein nicht definiertes Datenwort
1A15H	–	Transferfehler bei TDD auf ein nicht definiertes Datenwort
1A16H	–	Ladefehler bei LDR auf ein nicht definiertes Datenwort
1A17H	–	Ladefehler bei LDL auf ein nicht definiertes Datenwort
1A18H	–	Ladefehler bei LDW auf ein nicht definiertes Datenwort
1A19H	–	Ladefehler bei LDD auf ein nicht definiertes Datenwort

Sonstige Laufzeitfehler (OB 31)

Hierzu gehören alle Laufzeitfehler, die nicht einer der vorherigen Laufzeitfehlerarten (Transferfehler oder Aufruf eines nicht geladenen Bausteins) zugeordnet werden können.

Beim Erkennen eines dieser Laufzeitfehler ruft das Systemprogramm den Organisationsbaustein **OB 31** auf. Der den Fehler verursachende Befehl (bzw. die Sonderfunktion) wird nicht weiterbearbeitet. Wenn der OB 31 nicht geladen ist, geht die CPU in den Stoppzustand.
Soll die Programmbearbeitung bei Auftreten eines der unten aufgeführten Fehler weiterlaufen, genügt die Bausteinende-Anweisung BE im OB 31.

Bei Aufruf des OB 31 stehen in AKKU 1 und AKKU 2 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Fehleranzeigen von verschiedenen Operationen, OB 254/255 und OB 250

Tabelle 5-22 LZF – Sonstige Laufzeitfehler/Teil 1

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A21H	–	E DB, EX DX: Datenbaustein existiert bereits
1A22H	–	E DB, EX DX: unzulässige Anzahl Datenwörter (< 1 oder > 4091)
1A23H	–	E DB, EX DX: Speicherplatz im RAM reicht nicht aus
1A25H	–	BI: unzulässiger Parameter im AKKU 1 (<1 oder >125)
1A29H	–	Klammerstackunter- oder -überlauf nach ’U(’, ’O(’, ’)’
1A2AH	–	A DB, AX DX: Bausteinlänge im Datenbausteinkopf ist zu klein (Länge <5 Wörter)
1A2BH	–	Funktionsbaustein ist mit falscher PG-Software geladen
1A2CH	–	ACR: Kachelnummer in AKKU-1-L ist unzulässig (> 255)
1A31H	–	OB 254 bzw. OB 255 (Verschieben) oder OB 250: Ziel-Datenbaustein ist bereits im DB-RAM vorhanden
1A32H	–	OB 254 bzw. OB 255 (Duplizieren): Ziel-Datenbaustein ist bereits im DB-RAM vorhanden
1A33H	–	OB 254 bzw. OB 255 oder OB 250: Speicherplatz im DB-RAM reicht nicht aus

Fehleranzeigen von OB 182

Tabelle 5-23 LZF – Sonstige Laufzeitfehler/Teil 2 (Anzeigen von OB 182)

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A34H	0001H	Beschreibung des Datenfeldes ist unzulässig
1A34H	0100H	Adreßbereichs-Typ ist unzulässig
1A34H	0101H	Datenbaustein-Nr. ist unzulässig
1A34H	0102H	"Nummer des ersten Parameterwortes" ist unzulässig
1A34H	0200H	"Quelldatenbaustein-Typ" ist unzulässig
1A34H	0201H	"Quelldatenbaustein-Nummer" ist unzulässig
1A34H	0202H	"Nummer des ersten zu übertragenden Datenwortes in der Quelle" ist unzulässig
1A34H	0203H	Als Länge des Quelldatenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen
1A34H	0210H	"Zieldatenbaustein-Typ" ist unzulässig
1A34H	0211H	"Zieldatenbaustein-Nummer" ist unzulässig
1A34H	0212H	"Nummer des ersten zu übertragenden Datenwortes im Ziel" ist unzulässig
1A34H	0213H	Als Länge des Zieldatenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen
1A34H	0220H	"Anzahl zu übertragender Datenworte" ist unzulässig (= 0oder > 4091)
1A34H	0221H	Quelldatenbaustein ist zu kurz
1A34H	0222H	Zieldatenbaustein ist zu kurz
1A34H	0223H	Zieldatenbaustein ist im EPROM gespeichert

Fehleranzeigen verschiedener Sonderfunktions-OBs

Die nachfolgende Tabelle enthält Anzeigen von OB 110,OB 121, OB 122, OB 221, OB 240, OB 241, OB 242 un d OB 250.

Tabelle 5-24 LZF – Sonstige Laufzeitfehler/Teil 3

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A35H	–	OB 250: Nummer des Übergabebausteins ist unzulässig
1A36H	–	OB 250: unterschiedliche Länge bei DB x und DB x+ 1 bzw. DX x und DX x+ 1
1A3AH	–	OB 221: unzulässiger Wert für die neue Zykluszeit (Zykluszeit < 1 ms oder > 13000 ms)
1A3BH	–	OB 223: Anlaufarten der am Mehrprozessorbetrieb beteiligten CPUs sind unterschiedlich
1A41H	–	OB 240, OB 241 oder OB 242: Schieberegister- oder Datenbaustein-Nummer ist unzulässig (Nr. < 192 oder > 255)
1A42H	–	OB 241: Schieberegister ist nicht initialisiert
1A43H	–	OB 240: Speicherplatz im DB-RAM reicht nicht aus
1A44H	–	OB 240: Datenwort DW 0 des Datenbausteins hat nicht den Inhalt ’0’
1A45H	–	OB 240: Schieberegisterlänge in DW 1 is unzulässig (nicht zwischen 2 und 256)
1A46H	–	OB 240: Zeigerposition ist unzulässig oder Zeigeranzahl ist > 5
1A47H	–	OB 120: Wert in AKKU 1 oder AKKU-2-L ist unzulässig
1A48H	–	OB 122: Wert in AKKU 1 oder AKKU-2-L ist unzulässig
1A49H	–	OB 110: Wert in AKKU 1 oder AKKU-2-L ist unzulässig
1A4AH	–	OB 121: Wert in AKKU 1 oder AKKU-2-L ist unzulässig
1A4BH	–	OB 123: Wert in AKKU 1 ist unzulässig

Fehleranzeigen von OB 150

Tabelle 5-25 LZF – Sonstige Laufzeitfehler/Teil 4 (Anzeigen von OB 150)

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A4CH	0001H	Funktionsnummer ist unzulässig (=0 oder >2)
1A4CH	0100H	Adreßbereichs-Typ ist unzulässig
1A4CH	0101H	Datenbaustein-Nr. ist unzulässig
1A4CH	0102H	"Nummer des ersten Datenfeldwortes" ist unzulässig
1A4CH	0103H	als Länge des Datenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen
1A4CH	0201H	Jahresangabe im Datenfeld ist unzulässig
1A4CH	0202H	Monatsangabe im Datenfeld ist unzulässig
1A4CH	0203H	Monatstagangabe im Datenfeld ist unzulässig
1A4CH	0204H	Wochentagangabe im Datenfeld ist unzulässig
1A4CH	0205H	Stundenangabe im Datenfeld ist unzulässig
1A4CH	0206H	Minutenangabe im Datenfeld ist unzulässig
1A4CH	0207H	Sekundenangabe im Datenfeld ist unzulässig
1A4CH	0208H	Wert "1/100 Sekunde" im Datenfeld ist ungleich 0
1A4CH	0209H	Datenfeldwort 3/ Bit 0 bis 3 ist ungleich 0
1A4CH	020AH	Stunden-Format ist ungleich der Einstellung bei OB 151

Fehleranzeigen von OB 151, OB 152 und OB 153

Tabelle 5-26 LZF – Sonstige Laufzeitfehler/Teil 5 (Anzeigen von OB 151, OB 152und OB 153)

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
Anzeigen von OB 151		
1A4DH	0001H	Funktionsnummer ist unzulässig (=0 oder >2)
1A4DH	0100H	Adreßbereichs-Typ ist unzulässig
1A4DH	0101H	Datenbaustein-Nr. ist unzulässig
1A4DH	0102H	"Nummer des ersten Datenfeldwortes" ist unzulässig
1A4DH	0103H	als Länge des Datenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen
1A4DH	0201H	Jahresangabe im Datenfeld ist unzulässig
1A4DH	0202H	Monatsangabe im Datenfeld ist unzulässig
1A4DH	0203H	Monatstagangabe im Datenfeld ist unzulässig
1A4DH	0204H	Wochentagangabe im Datenfeld ist unzulässig
1A4DH	0205H	Stundenangabe im Datenfeld ist unzulässig
1A4DH	0206H	Minutenangabe im Datenfeld ist unzulässig
1A4DH	0207H	Sekundenangabe im Datenfeld ist unzulässig
1A4DH	0208H	Wert "1/100 Sekunde" im Datenfeld ist ungleich 0
1A4DH	0209H	Auftragsart im Datenfeld ist unzulässig (> 7)
1A4DH	020AH	Stunden-Format ist ungleich der Einstellung bei OB 150
Anzeigen von OB 152		
1A4EH	0001H	Funktionsnummer ist unzulässig (ungleich 0 bis 3 oder ungleich 8 oder ungleich 15)
Anzeigen von OB 153		
1A4FH	0001H	Funktionsnummer ist unzulässig (=0 oder <0)
1A4FH	0002H	Verzögerungszeit unzulässig

Fehleranzeigen von verschiedenen Systemoperationen

Tabelle 5-27 LZF – Sonstige Laufzeitfehler/Teil 6 (Anzeigen von verschiedenen Systemoperationen)

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1A50H	–	LRW, TRW: Die errechnete Speicheradresse <BR+ Konstante> liegt nicht im Bereich "0 .. EDFFH" 1)
1A51H	–	LRD, TRD: Die errechnete Speicheradresse <BR+ Konstante>liegt nicht im Bereich "0 .. EDFEH" 1)
1A52H	–	TSG, LB GB, LW GW, TB GB, TW GW: Die errechnete Linearadresse <BR+Konstante> liegt nicht im Bereich "0 .. EFFFH"
1A53H	–	LB GW, LW GD, TB GW, TW GD: Die errechnete Linearadresse <BR+Konstante>liegt nicht im Bereich "0 .. EFFEH"
1A54H	–	LB GD, TB GD: Die errechnete Linearadresse <BR+Konstante>liegt nicht im Bereich "0 .. EFFCH"
1A55H	–	TSC, LB CB, LW CW, TB CB, TW CW: Die errechnete Kacheladresse <BR+Konstante>liegt nicht im Bereich "F400H .. FBFFH"
1A56H	–	LB CW, LW CD, TB CW,TW CD: Die errechnete Kacheladresse <BR+Konstante> liegt nicht im Bereich "F400H .. FBFEH"
1A57H	–	LB CD, TB CD: Die errechnete Kacheladresse <BR+Konstante>liegt nicht im Bereich "F400H .. FBFCH"

Fehlerkennung AKKU-1-L AKKU-2-L	**Erläuterung**
Fortsetzung der Tabelle 5-27:	
1A58H –	TNW, TNB: Der Quellblock liegt nicht vollständig in einem dieser Bereiche: 0000 .. 7FFF Anwenderspeicher [1)] 8000 .. DD7F Datenbaustein-RAM DD80 .. E3FF DB 0 E400 .. E7FF S-Merker E800 .. EDFF Systemdaten (BA, BB, BS, BT, Z, T) EE00 .. EFFF Merker, Prozeßabbild F000 .. FFFF Peripherie
1A59H –	TNW, TNB: Der Zielblock liegt nicht vollständig in einem dieser Bereiche: 0000 .. 7FFF Anwenderspeiche [1)] 8000 .. DD7F Datenbaustein-RAM DD80 .. E3FF DB 0 E400 .. E7FF S-Merker E800 .. EDFF Systemdaten (BA, BB, BS, BT, Z, T) EE00 .. EFFF Merker, Prozeßabbild F000 .. FFFF Peripherie

[1)] siehe Kapitel 9

5.6.3 ADF (Adressierfehler)

Ein Adressierfehler tritt auf, wenn mit einer STEP-5-Operation ein Ein- oder Ausgang im Prozeßabbild angesprochen wird, dem zum Zeitpunkt des letzten NEUSTARTs keine Peripheriebaugruppe zugeordnet war (Baugruppe war nicht gesteckt, defekt oder nicht im Datenbaustein DB 1 der CPU angegeben).

OB 25

Das Systemprogramm unterbricht nun die Bearbeitung des Anwenderprogramms und ruft den Organisationsbaustein **OB 25** auf. Nach der Bearbeitung des im OB 25 enthaltenen Programms wird mit dem nächsten Befehl des unterbrochenen Programms fortgefahren. Die STEP-5-Anweisung, die den ADF verursacht hat, wurde zuvor ausgeführt, jedoch mit einem undefinierten Ein- oder Ausgangswert!
Wenn der OB 25 nicht programmiert ist, geht die CPU beim Auftreten eines Adressierfehlers in den Stoppzustand, es sei denn, Sie haben für diesen Fall eine Fortsetzung der Programmbearbeitung im Datenbaustein DX 0 festgelegt.
Die Adressierfehler-Überwachung kann durch entsprechende Programmierung des DX 0 auch ganz unterdrückt werden.

Fehleranzeigen

Als Fehleranzeigen werden vom Systemprogramm übergeben:

AKKU-1-L = 1E40H

AKKU-2-L = ADF-Adresse

5.6.4 QVZ (Quittungsverzug)

Ein Quittungsverzug tritt auf, wenn sich eine Ein- oder Ausgabebaugruppe nach einer Adressierung innerhalb einer bestimmten Zeit nicht mit dem RDY-Signal (Ready) zurückmeldet. Die Ursache des Quittungsverzugs kann ein Defekt auf der Peripheriebaugruppe sein oder das Ziehen der Baugruppe aus dem AG während des Betriebs.

Folgende Quittungsverzugfehler unterbrechen die Anwenderprogrammbearbeitung und rufen einen entsprechenden Organisationsbaustein auf:

QVZ bei Direktzugriff über S5-Bus

Quittungsverzug im Anwenderprogramm bei Direktzugriff über den S5-Bus auf CP, IP, KOR oder auf eine Peripheriebaugruppe (z.B. mit Lade- und Transferbefehlen L/T P...bzw. Q...):

OB 23

Das Systemprogramm ruft den Organisationsbaustein **OB 23** auf, wenn dieser geladen ist.

Fehleranzeigen

In den AKKUs 1 und 2 stehen dabei zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern

AKKU-1-L = 1E23H

AKKU-2-L = QVZ-Adresse

QVZ-Adresse

Die QVZ-Addresse weist auf dasjenige Peripheriebyte, welches als **erstes** einen QVZ erzeugt hat. In der Regel ist dies das Byte mit der niedrigsten Adresse bei Peripheriebefehlen.

Eine Ausnahme hiervon sind QVZ-Addressen, die bei den Befehlen TNB/TNW im Fall eines Quittungsverzuges geliefert werden: Da diese Befehle dekrementierend arbeiten, zeigt in diesem Fall die QVZ-Adresse auf das Byte mit der höchsten Addresse, das bei einem Blocktransfer den QVZ ausgelöst hat.

QVZ bei PAE/PAA-Aktualisierung und Transfer der Koppelmerker

OB 24

Das Systemprogramm ruft den Organisationbaustein **OB 24** auf. In den AKKUs 1 und 2 stehen dabei zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-28 QVZ – Anzeigen bei Aufruf des OB 24

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
1E25H	yyyyH	Quittungsverzug bei der Ausgabe des Prozeßabbildes der digitalen Ausgänge yyyy =Adresse des nicht quittierten Ausgangsbytes
1E26H	yyyyH	Quittungsverzug beim Aktualisieren des Prozeßabbildes der digitalen Eingänge yyyy =Adresse des nicht quittierten Eingangsbytes
1E27H	yyyyH	Quittungsverzug beim Aktualisieren der Koppelmerker-Ausgänge yyyy =Adresse des nicht quittierten Koppelmerkerbytes
1E28H	yyyyH	Quittungsverzug beim Aktualisieren der Koppelmerker-Eingänge yyyy =Adresse des nicht quittierten Koppelmerkerbytes

Hinweis
Wenn die aufgerufenen Organisationsbausteine **nicht programmiert** sind, wird die Bearbeitung des Anwenderprogramms **fortgesetzt**.

Bei einem aufgetretenen Quittungsverzug liest die CPU "ersatzweise" den Wert "00H" ein und arbeitet, falls der QVZ quittiert wird, mit diesem Wert weiter.

Durch einen Quittungsverzug wird die Laufzeit des STEP-5-Anwenderprogramms verlängert.

STOP bei QVZ

Wenn der Quittungsverzug zum STOP der CPU führen soll, muß im OB 23 bzw. 24 der Stoppbefehl STP programmiert sein.
Durch entsprechende Programmierung des DX 0 können Sie im Falle eines QVZ auch bei nicht programmierten OB 23/24 einen Systemstopp veranlassen.

5.6.5 ZYK (Zykluszeitfehler)

Die Zykluszeit umfaßt die gesamte Zeitdauer einer Bearbeitung des zyklischen Programms. Eine Überschreitung der in der CPU eingestellten Zyklusüberwachungszeit kann ausgelöst werden z.B. durch fehlerhafte Programmierung, durch eine Programmschleife in einem Funktionsbaustein, durch Ausfall des Taktgenerators oder durch Systemleistungen wie z.B. Prozeßabbildaktualisierung in Verbindung mit langen Programmen.

OB 26

Wenn eine Zykluszeitüberschreitung auftritt, unterbricht das Systemprogramm die Bearbeitung des Anwenderprogramms und ruft den Organisationsbaustein **OB 26** auf, wenn dieser geladen ist. Die Überwachungszeit wird dabei neu gestartet (getriggert). Falls die Überwachungszeit erneut abläuft, bevor der OB 26 zu Ende bearbeitet ist, geht die CPU mit Doppelfehler (DOPP-FE) in den Stoppzustand.

Zyklusüberwachungszeit

Die Zyklusüberwachungszeit ist variabel (1 bis 13000 ms) und nachtriggerbar (siehe oben). Unabhängig von der Zykluszeit wird 100 ms nach Ablauf der Zykluszeit BASP aktiv gesetzt, wenn der OB 26 zu diesem Zeitpunkt noch nicht zu Ende bearbeitet ist.

Die Zyklusüberwachungszeit können Sie individuell vorgeben durch einen Eintrag im DX 0 oder durch Aufruf des Sonderfunktions-Organisationsbausteins OB 221.

Im zyklischen Programm kann die Zyklusüberwachungszeit durch einen Aufruf des Sonderfunktions OB 222 "nachgetriggert" werden.

STOP bei nicht geladenem OB 26

Wenn der OB 26 nicht programmiert ist, geht die CPU in den Stoppzustand. Soll dies nicht geschehen, so müssen Sie die Voreinstellung im DX 0 entsprechend ändern.

keine Fehlerkennungen

Bei Auftreten eines Zykluszeitfehlers werden **keine** Fehlerkennungen in AKKU 1 oder AKKU 2 übergeben!

5.6.6 WECK-FE (Weckfehler)

Wenn für einen bestimmten Weckalarm-OB eine erneute Anforderung auftritt, bevor seine letzte Anforderung vollständig bearbeitet ist, erkennt das Systemprogramm einen Weckfehler und ruft den Organisationsbaustein **OB 33** auf, wenn dieser geladen ist, oder die CPU geht in den Stoppzustand. Beachten Sie hierzu auch den Abschnitt 4.5.2.

In den AKKUs 1 und 2 hinterlegt das Systemprogramm zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern:

Tabelle 5-29 WECK-FE– Anzeigen

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung	
1001H	0016H	Weckfehler bei OB 10	(10 ms)
1001H	0014H	Weckfehler bei OB 11	(20 ms)
1001H	0012H	Weckfehler bei OB 12	(50 ms)
1001H	0010H	Weckfehler bei OB 13	(100 ms)
1001H	000EH	Weckfehler bei OB 14	(200 ms)
1001H	000CH	Weckfehler bei OB 15	(500 ms)
1001H	000AH	Weckfehler bei OB 16	(1 sec)
1001H	0008H	Weckfehler bei OB 17	(2 sec)
1001H	0006H	Weckfehler bei OB 18	(5 sec)

Hinweis
Die Kennung im AKKU 2 ist die Ebenenkennung des fehlererzeugenden Weckalarms.

Ist der OB 33 nicht programmiert, geht die CPU in den Stoppzustand. Durch entsprechende Programmierung im **DX 0** können Sie bei aufgetretenem Weckfehler und nicht programmiertem OB 33 die Programmbearbeitung jedoch weiterlaufen lassen.

Ein erneuter Aufruf der bereits aktivierten Fehlerprogrammbearbeitungsebene "Weckfehler" führt nicht zu einem Doppelfehler (DOPP)!

5.6.7 REG-FE (Reglerfehler)

Ein Fehler beim Bearbeiten der vom Systemprogramm unterstützten Standard-Funktionsbausteine der Reglerstruktur R 64 wird als Reglerfehler erkannt.

> **Hinweis**
> Weckfehler werden vom Systemprogramm immer dann erkannt, wenn ein bestimmter Weckalarm-OB nicht innerhalb seines Zeitrasters (z. B. der OB 13 innerhalb von 100 ms) begonnen und zu Ende bearbeitet wurde. Eine fehlerhafte Bearbeitung des Regelungsprogramms wird jedoch **erst bei Aufruf** der Programmbearbeitungsebene REGELUNG erkannt und im USTACK angezeigt.

OB 34

Bei Auftreten eines Reglerfehlers wird die Programmbearbeitungsebene REGELUNG verlassen und die Ebene REGLER-FEHLER (EBENE: 001CH) mit dem Organisationsbaustein **OB 34** aufgerufen.
Die weitere Reaktion der CPU hängt ab von der Programmierung des OB 34:

- Wenn der OB 34 nicht programmiert ist, geht die CPU in STOP. Durch Ausgabe des USTACKs läßt sich die Fehlerursache ermitteln.
- Wenn der OB 34 programmiert ist, wird das darin enthaltene STEP-5-Programm (z. B. Auswertung von AKKU 1 und 2, davon abhängig die Fehlerbehandlung) bearbeitet. Im Anschluß daran wird die Reglerbearbeitung an der unterbrochenen Stelle fortgesetzt.

Reaktion auf Reglerfehler

Sollen Reglerfehler grundsätzlich ignoriert werden, so genügt ein Bausteinenende-Befehl BE im OB 34.

Soll die Reglerbearbeitung bei aufgetretenem Reglerfehler auch bei nicht programmiertem OB 34 fortgesetzt werden, müssen Sie die Voreinstellung im DX 0 entsprechend ändern.

Bei Aufruf des OB 34 stehen in den AKKUs 1 und 2 zusätzliche Informationen, die den aufgetretenen Fehler näher erläutern.

Tabelle 5-30 REG-FE– Anzeigen

Fehlerkennung AKKU-1-L	AKKU-2-L	Erläuterung
0801H	DByyH	Abtastzeitfehler yy = Nummer des betreffenden Regler-Datenbausteins
0802H	DByyH	Regler-Datenbaustein nicht geladen yy = Nummer des nicht geladenen Datenbausteins
0803H	FByyH	Regler-Funktionsbaustein nicht geladen yy = Nummer des nicht geladenen Funktionsbausteins
0804H	FByyH	Regler-Funktionsbaustein nicht erkannt yy = Nummer des nicht erkannten Funktionsbausteins
0805H	FByyH	Regler-Funktionsbaustein mit falscher PG-Software geladen yy = Funktionsbaustein-Nr.
0806H	DByyH	Falsche Regler-Datenbaustein-Länge yy = Datenbaustein-Nr.
0880H	00yyH	Quittungsverzug (QVZ) während der Reglerbearbeitung yy = Nr. des E/A-Bytes, das QVZ verursacht hat.

Eintrag in Steuerbit-Maske

In allen 7 Fehlerfällen wird am Programmiergerät in den Steuerbits die Fehlerkennung **REG-FE** angekreuzt. Wenn Sie ein PG ohne S5-DOS-Betriebssystem verwenden, ist die vorletzte Stelle in der unteren Zeile der Steuerbits-Maske zwar nicht bezeichnet, wird jedoch ebenso angekreuzt. In der USTACK-Maske der Ebene REGELUNG ist als Störungsursache **REG** angekreuzt.

Abtastzeitfehler

Nach Ablauf der vorgegebenen Abtastzeit wird das zyklische Programm an der nächsten **Bausteingrenze** abgebrochen und die Reglerbearbeitung eingeschoben. Nun ist es möglich, daß die Bearbeitung "langer" Bausteine zu viel Zeit in Anspruch nimmt und in der Folge die Reglerbearbeitung "außer Tritt" gerät: Es liegt ein Abtastzeitfehler vor.

Ein Abtastzeitfehler kann wie die übrigen Reglerfehler behandelt werden (wie auf der vorigen Seite beschrieben) **oder** über eine Maske unterdrückt werden. In diesem Fall wird bei Auftreten eines Abtastzeitfehlers die Programmbearbeitung nicht unterbrochen.

Beachten Sie dazu die Beschreibung "Kompaktregelung im R-Prozessor des AG S5-135U" /13/

Ein Abtastzeitfehler kann eventuell verhindert werden, indem Sie die Voreinstellung im DX 0 "Bearbeitung des Regler- und Prozeßalarms an Bausteingrenzen" in "Bearbeitung des Regler- und Prozeßalarms an Befehlsgrenzen" abändern.

5.6.8 ABBR (Abbruch)

Wenn im Betriebszustand RUN der Stoppzustand angefordert wird durch

- Betätigen des Betriebsartenschalters an der CPU von RUN auf STOP,
- PG-Online-Funktion AG-STOP,
- Schalter am Koordinator auf STOP (im Mehrprozessorbetrieb),

so ruft das Systemprogramm den **OB 28** auf, wenn dieser geladen ist. Nach der Bearbeitung des OB 28 geht die CPU in den **Stoppzustand**.

> **Hinweis**
> Der Übergang in den Stoppzustand erfolgt unabhängig davon, ob und wie der OB 28 programmiert ist.

keine Fehlerkennungen

Es werden **keine** Fehlerkennungen in AKKU 1 oder AKKU 2 übergeben!

5.6.9 Kommunikationsfehler (FE-3)

Treten auf der zweiten seriellen Schnittstelle bei Rechnerkopplung RK 517, Datenübertragung mit Prozedur 3964/3964R, Datenübertragung mit "offenem Treiber" oder bei Datenübertragung mit SINEC L1 Störungen auf, so ruft das Systemprogramm den Organisationsbaustein **OB 35** auf und übergibt in AKKU 1 zusätzliche Informationen, die die aufgetretenen Störungen näher erläutern.

Reaktion bei nicht geladem OB 35

Haben Sie keinen OB 35 programmiert, so erfolgt **keine** Reaktion des Systemprogramms und die CPU geht **nicht** in den Stoppzustand. Das ist die Standardeinstellung.
Soll beim Auftreten eines Schnittstellenfehlers die CPU auch bei nicht vorhandenem OB 35 in den Stoppzustand gehen, so müssen Sie die Voreinstellung im DX 0 entsprechend ändern.

Fehlerinformation in AKKU 1

Alle 100 ms prüft das Systemprogramm, ob Kommunikationsfehler an der zweiten seriellen Schnittstelle aufgetreten sind. Ist dies der Fall, so hinterlegt das Systemprogramm im AKKU 1 Fehlerinformationen. Ist der OB 35 vorhanden, so wird er vom Systemprogramm aufgerufen und die Fehlerinformation in AKKU 1 übergeben.

Es können Fehlernummern zu maximal drei Störungsursachen beim Aufruf des OB 35 übergeben werden. Liegen gleichzeitig mehr als drei Störungsursachen vor, so wird dies durch eine spezielle Überlaufkennung angezeigt.

Aufbau der Fehlerinformation in AKKU 1

	31							24	23 18	15 8	7 0
AKKU 1	0	0	0	0	F	U	B	0	Fehlernummer 1	Fehlernummer 2	Fehlernummer 3

F = ’0’, wenn kein Fehlereintrag im Fehlerbereich
= ’1’, wenn Fehler in Fehlerbereich eingetragen

U = ’0’, wenn kein Fehlerüberlauf (maximal drei Einträge)
= ‘1‘, wenn Fehlerüberlauf (mehr als drei Einträge)

B = ’0’, wenn kein BREAK auf der Schnittstelle
= ’1’, wenn BREAK auf der Schnittstelle

BREAK

Bei BREAK auf der Schnittstelle wird der OB 35 nur zu Beginn des BREAK-Zustandes aufgerufen.

Fehlernummer 1 bis Fehlernummer 3

Hier werden maximal 3 Fehlernummern zu den auf der Schnittstelle erkannten Störungen eingetragen und zwar in der Reihenfolge, wie sie vom System erkannt werden.

Bedeutung der Fehlernummern

Die Bedeutung der Fehlernummern sowie weitere Informationen zur Behandlung von Schnittstellenfehlern entnehmen Sie bitte dem Handbuch "Kommunikation" /14/.

Integrierte Sonderfunktionen 6

Inhalt von Kapitel 6

Integrierte Sonderfunktionen

6

Das nachfolgende Kapitel schildert Ihnen, welche integrierten Sonderfunktionen das Systemprogramm enthält, wo Sie diese anwenden können und wie Sie die Sonderfunktions-OBs aufrufen und parametrieren müssen.
Ferner erfahren Sie, wie Sie Fehler bei der Bearbeitung einer Sonderfunktion erkennen und per Programm bearbeiten können.

6.1 Einführung

Das Betriebssystem der CPU 928B bietet Ihnen Sonderfunktionen an, die Sie bei Bedarf mit einem bedingten (SPB OB x) oder einem unbedingten (SPA OB x) Bausteinaufruf aufrufen können. Für diese Sonderfunktionen sind die Organisationsbausteine OB 100 bis 255 reserviert.

Diese Funktionen werden als **Integrierte** Sonderfunktionen bezeichnet, da sie ein fester Bestandteil des Systemprogramms sind. Als Anwender können Sie diese Sonderfunktionen zwar aufrufen, jedoch nicht lesen oder ändern.

Die nachfolgende Tabelle gibt Ihnen eine Übersicht der vorhandenen Sonderfunktionen.

Tabelle 6-1 Übersicht der vorhandenen Sonderfunktionen

Baustein	**Funktion**	**siehe Abschnitt/Seite**
OB 110 OB 111 OB 112 OB 113	Zugriff auf Anzeigenbyte AKKU 1, 2, 3 und 4 löschen AKKU-Roll-Up AKKU-Roll-Down	6.2/6 - 6 6.3/6 - 13 6.4/6 - 14 "
OB 120 OB 121 OB 122 OB 123	"Alarme gemeinsam sperren" ein-/auschalten "Weckalarme einzeln sperren" ein-/ausschalten "Alarme gemeinsamverzögern" ein-/auschalten "Weckalarme einzeln verzögern" ein-/ausschalten	6.5/6 - 16 6.6/6 - 19 6.7/6 - 22 6.8/6 - 25
OB 150 OB 151	Systemzeit stellen/lesen Zeit für uhrzeitgesteuerten Weckalarm stellen/lesen	6.9/6 - 28 6.10/6 - 33
OB 152	Zyklusstatistik	6.11/6 - 40
OB 153	Zeit für Verzögerungsalarm stellen/lesen (ab Version -3UB12)	6.12/6 - 48
OB 160 bis 163	Zählschleife	6.13/6 - 51
OB 170	Bausteinstack (BSTACK) lesen	6.14/6 - 53
OB 180 OB 181 OB 182	variabler Datenbausteinzugriff Datenbaustein (DX/DX) testen Datenbereich kopieren	6.15/6 - 58 6.16/6 -62 6.17/6 - 65
OB 190, 192 OB 191, 193	Merker in Datenbausteine übertragen Datenblöcke in Merkerbereich übertragen	6.18/6 - 68 6.19/6 - 71
OB 200 [1], 202 [1] OB 203, 204 [1], 205	Funktionen zur Mehrprozessor-Kommunikation	6.20/6 -77
OB 216 bis 218	Kachelzugriffe	6.21/6 -78

Baustein	Funktion	siehe Abschnitt/Seite
Fortsetzung der Tabelle 6-1:		
OB 220	Vorzeichenerweiterung	6.22/6 - 90
OB 221 [2)] OB 222 OB 223 OB 224 [2)] OB 226 OB 227 OB 228	Zyklusüberwachungszeit einstellen Zyklusüberwachungszeit neu starten Anlaufarten vergleichen Koppelmerker blockweise übertragen Wort aus Systemprogramm lesen Quersumme des Systemprogramms lesen Statusinformation einer Programmbearbeitungs-ebene lesen	6.23/6 - 91 6.24/6 - 92 6.25/6 - 93 6.26/6 - 94 6.27/6 - 95 6.28/6 - 96 6.29/6 - 98
OB 230 bis 237 [1)]	Funktionen für Standard-Funktionsbausteine	6.30/6 - 100
OB 240 Ob 241 OB 242	Schieberegister initialisieren Schieberegister bearbeiten Schieberegister löschen	6.31.2/6 - 105 6.31.3/6 - 108 6.31.4/6 - 109
OB 250 [1)] OB 251 [1)]	Regelung: PID-Algorithmus initialisieren Regelung: PID-Algorithmus bnearbeiten	6.32.3/6 - 118 6.32.4/6 - 119
OB 254, 255 [1)]	einen DB- bzw. DX-Datenbaustein kopieren/duplizieren	6.33/6 - 125

[1)] Sonderfunktionen mit Pseudobefehlen (langlaufend)

[2)] Nutzen Sie an Stelle dieser Sonderfunktions-Organisationsbausteine die Parametrierung im Datenbaustein DX 0 (siehe Kapitel 7).

Schnittstellen

Als Schnittstellen zu den Sonderfunktionen stehen Ihnen bei der Programmierung zur Verfügung:

Bausteinaufruf

- Aufruf mit bedingtem/unbedingtem Bausteinaufruf SPB .. / SPA ..

Parameter

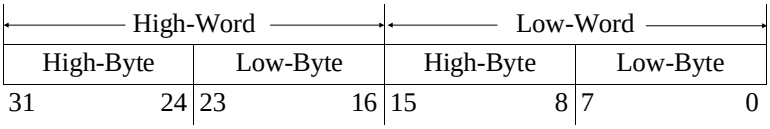
- Parameter zur Voreinstellung über AKKU 1 und evtl. AKKU 2 und/oder Speicherzellen.

  Unter dem Begriff **Parameter** sind in der folgenden Beschreibung der einzelnen Sonderfunktionen alle Daten aufgelistet, die die CPU benötigt, um die Sonderfunktion korrekt ausführen zu können. Vor Aufruf der Sonderfunktion im STEP-5-Programm müssen Sie diese Daten in die Akkumulatoren oder in die jeweils angegebenen Speicherzellen laden.

Akku-Schreibweise

Bei den Angaben zur Parametrierung der einzelnen Sonderfunktions-Organisationsbausteine berücksichtigen Sie bitte folgende Schreibweise:

AKKU 1:	AKKU 1,	**32 bit**
AKKU-1-L:	AKKU 1, Low-Wort,	**16 bit**
AKKU-1-LL:	AKKU 1, Low-Wort, Low-Byte,	**8 bit**
AKKU-1-LH:	AKKU 1, Low-Wort, High-Byte,	**8 bit**



Fehlerbearbeitung

Tritt bei der Bearbeitung der aufgerufenen Sonderfunktion ein Fehler auf, so reagiert das Systemprogramm darauf mit einer speziellen Fehlerreaktion.

Hinsichtlich dieser Fehlerreaktion des Systemprogramms können zwei Gruppen von Sonderfunktionen unterschieden werden.

Fehler-OB, Akku-Anzeigen

Zur Gruppe 1 zählen alle Sonderfunktionen, bei denen im Fehlerfall ein **Fehler-Organisationsbaustein** (Fehler-OB) aufgerufen wird, in dem Sie das weitere Verhalten der CPU festlegen können. Diese Fehler-OB sind der OB 19, der OB 30 und der OB 31. In AKKU 1 und bei einigen Sonderfunktionen zusätzlich in AKKU 2 (siehe Kapitel 5.6.1 und 5.6.2) werden dem Fehler-OB Kennungen übergeben, die den aufgetretenen Fehler näher erläutern.

Stößt die CPU bei der Bearbeitung einer dieser Sonderfunktionen z. B. auf eine falsche Parametrierung, so erkennt sie einen Laufzeitfehler und ruft den OB 31 auf. Ist z. B. die aufgerufene Sonderfunktion nicht vorhanden, erkennt die CPU einen Befehlscodefehler und versucht, den OB 30 aufzurufen. Wird bei einigen Sonderfunktionen in den Aufrufparametern auf einen Datenbaustein verwiesen und ist dieser Datenbaustein nicht geladen, so versucht die CPU den OB 19 aufzurufen.
Falls die Fehler-OB 30 bzw. 31 nicht geladen sind oder einen STP-Befehl enthalten, geht die CPU in den Stoppzustand. In den Steuerbits und im USTACK ist LZF bzw. BCF angekreuzt. In den Akkumulatoren der Fehlerbearbeitungsebene sind Fehlerkennungen hinterlegt, die den Fehler näher beschreiben. Falls der OB 19, der OB 30 oder der OB 31 geladen sind (und keine STP-Operation enthalten), wird das Anwenderprogramm nach der Bearbeitung des betreffenden OBs mit dem nächsten Befehl fortgesetzt. In diesem Fall sind die Akkumulatoren unverändert.

VKE ,
ANZ 0/ANZ 1

Bei einigen Sonderfunktionen werden für das Anzeigen von sonderfunktionsspezifischen Fehlern das VKE oder die Anzeigen ANZ 0/ANZ 1 beeinflußt.
Wenn bei der Bearbeitung dieser Sonderfunktionen ein Fehler auftritt, wird in den meisten Fällen das VKE gesetzt (VKE = 1). Sie können in Ihrem STEP-5-Programm bei diesen Sonderfunktionen mit einer SPB-Operation (Springe bedingt) das VKE auswerten und damit auf einen Fehler reagieren.

Bei manchen Sonderfunktionen werden die Ergebnisanzeigen ANZ 0 und ANZ 1 durch die Bearbeitung der Sonderfunktion beeinflußt. Diese Anzeigen können Sie in Ihrem STEP-5-
Programm mit Vergleichsoperationen abfragen und somit ebenfalls auf einen Fehler reagieren.

Welche der geschilderten Fehlerreaktionen bei den einzelnen Sonderfunktions-OB ausgeführt werden, wird in den folgenden Unterkapiteln zu den Sonderfunktions-OB beschrieben.

Hinweis
Der Aufruf eines Sonderfunktions-OB mit der Operation "SPB OB" bzw. "SPA OB" wirkt nicht wie ein "echter" Bausteinwechsel, sondern wie eine STEP-5-Operation. Es werden **keine Alarme** eingeschachtelt (bei Voreinstellung "Unterbrechung an Bausteingrenzen")!

Sonderfunktionen mit Pseudobefehlsgrenzen

Einige der Sonderfunktionen sind langlaufende Sonderfunktionen, die sogenannte Pseudobefehlsgrenzen enthalten.
Dies bedeutet, daß die Ausführung der Sonderfunktion in mehreren Teilschritten erfolgt. Wenn nun während der Ausführung eines Teilschritts eine Unterbrechung (z. B . Weck- oder Prozeßalarm an Befehlsgrenzen) auftritt, so wird am Ende dieses Teilschritts an der Pseudobefehlsgrenze der entsprechende Organisationsbaustein eingeschachtelt.
Diejenigen Sonderfunktionen, die Pseudobefehlsgrenzen enthalten, sind in der Übersichtstabelle der integrierten Sonderfunktionen gekennzeichnet.

6.2 OB 110: Zugiff auf das Anzeigenbyte

Funktion

Mit Hilfe des OB 110 können Sie das Anzeigenregister mit dem Inhalt des AKKU 1 beschreiben, mit ’1’ maskieren oder mit ’0’ maskieren.

Belegung des AKKU 1 für den Zugriff auf das Anzeigenregister

31	7	6	5	4	3	2	1	0
*)	A 1	A 0	OV	OS	OR	STA	VKE	$\overline{\text{ERAB}}$
	Wort-Anzeigen				**Bit**-Anzeigen			

*) Die Bits 8 bis 31 sind für Erweiterungen reserviert und müssen beim Beschreiben des Anzeigenregisters gleich ’0’ sein. Sie müssen beim Auslesen des Anzeigenregisters ignoriert werden.

Parameter

1. AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1, 2 oder 3

2. AKKU 2

Neues Anzeigenbyte bzw. Maske

Funkt.-Nr. in AKKU-2L	**Inhalt von AKKU-1-L**		**Funktion**
	vorher	**nachher**	
1	neues Anzeigenbyte	neues Anzeigenbyte	Der Inhalt von AKKU 1 wird in das Anzeigenregister geladen.
2	Maske	neues Anzeigenbyte [1)]	Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Anzeigenregister gleich **’1’** gesetzt. Das neue Anzeigenbyte wird in AKKU 1 geladen.
3	Maske	neues Anzeigenbyte [1)]	Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Anzeigenregister gleich **’0’** gesetzt. Das neue Anzeigenbyte wird in AKKU 1 geladen.

[1)] Einschränkung: Der OB 110 selbst beeinflußt die Anzeigenbits vor der Maskierung entsprechend einem Baustein-Aufruf, d. h. er setzt OR = 0, STA = 1 und $\overline{\text{ERAB}}$ = 0. Die Maskierung erfolgt auf diese veränderten Bits.

Ergebnis

Nach Ablauf des OB 110 ist das Anzeigenbyte entsprechend Funktion und AKKU-1-Inhalt verändert.

Fehlerfälle

- Funktions-Nr. in AKKU-2-L ungleich 1, 2 oder 3.
- In AKKU 1 ist eines der Bits Nr 8 bis Nr. 31 gesetzt.

Im Fehlerfall wird der **OB 31** (Sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen ist in AKKU-1-L die Fehlerkennung 1A49H hinterlegt

Beispiel

Der OB 110 ist ein Hilfsmittel zum Testen der Operationen, die das Anzeigenregister auswerten oder beeinflussen. Seine Anwendung ist jedoch nicht hierauf begrenzt. Folgendes Beispiel soll einen weiteren Anwendungsfall zeigen.

```
Aufrufverteiler:

In Abhängigkeit vom Inhalt des Merkerbytes MB 0 soll eines von vier Teilpro-
grammen aufgerufen werden. Den vier Programmteilen werden die Bits M 0.0 bis
M 0.3 zugeordnet. Es darf immer nur eines dieser Bits gesetzt sein.

        :L      MB 0
        :SLW    4           M 0.0 bis M 0.3 um vier Bits nach links schieben
        :L      KB 1        Funktions-Nr. laden
        :TAK
        :SPA    OB 110
        :SPS    =M000       Sprung falls OS = 1
        :SPO    =M001       Sprung falls OV = 1
        :SPM    =M002       Sprung falls ANZ 0 = 1
        :SPP    =M003       Sprung falls ANZ 1 = 1
        :
        :                   falls kein Bit gesetzt
        :
        :BEA
        :
  M000  :                   falls M 0.0 = 1
        :
        :BEA
  M001  :                   falls M 0.1 = 1
        :
        :BEA
  M002  :                   falls M 0.2 = 1
        :
        :BEA
  M003  :                   falls M 0.3 = 1
        :
        :BEA
```

6.3 OB 111: AKKU 1, 2, 3 und 4 löschen

Funktion

Durch den einmaligen Aufruf des Sonderfunktions-Organisationsbausteins OB 111 können Sie die Inhalte der AKKUs 1 bis 4 auf einfache Weise löschen: Der OB 111 überschreibt alle vier Register mit ’0’.

Parameter

keine

Ergebnis

Die AKKUs 1 bis 4 (je 32 bit) sind gelöscht.

Fehlerfälle

keine

6.4 OB 112/113: AKKU-Roll-Up/AKKU-Roll-Down

Funktion

OB 112 und OB 113 bewirken ein "Rollen" der AKKU-Inhalte in aufsteigender bzw. absteigender Richtung:

- Der OB 112 (Roll Up) verschiebt den Inhalt von AKKU 1 in den AKKU 2, den Inhalt von AKKU 2 in den AKKU 3 usw...
- Der OB 113 (Roll Down) verschiebt die AKKU-Inhalte in entgegengesetzte Richtung: Inhalt von AKKU 1 in den AKKU 4, AKKU 4 in AKKU 3 usw.

Parameter

keine

Ergebnis

Die Bilder 6-1 und 6-2 zeigen die AKKU-Inhalte **vor** und **nach** dem Aufruf von OB 112 und 113.

> **Hinweis**
> Mit den STEP-5-Operationen ENT (Ergänzender Operationsvorrat) und TAK (Systemoperation) lassen sich die AKKU-Inhalte ebenfalls verschieben (siehe Abschnitt 3.4.3).

Fehlerfälle

keine

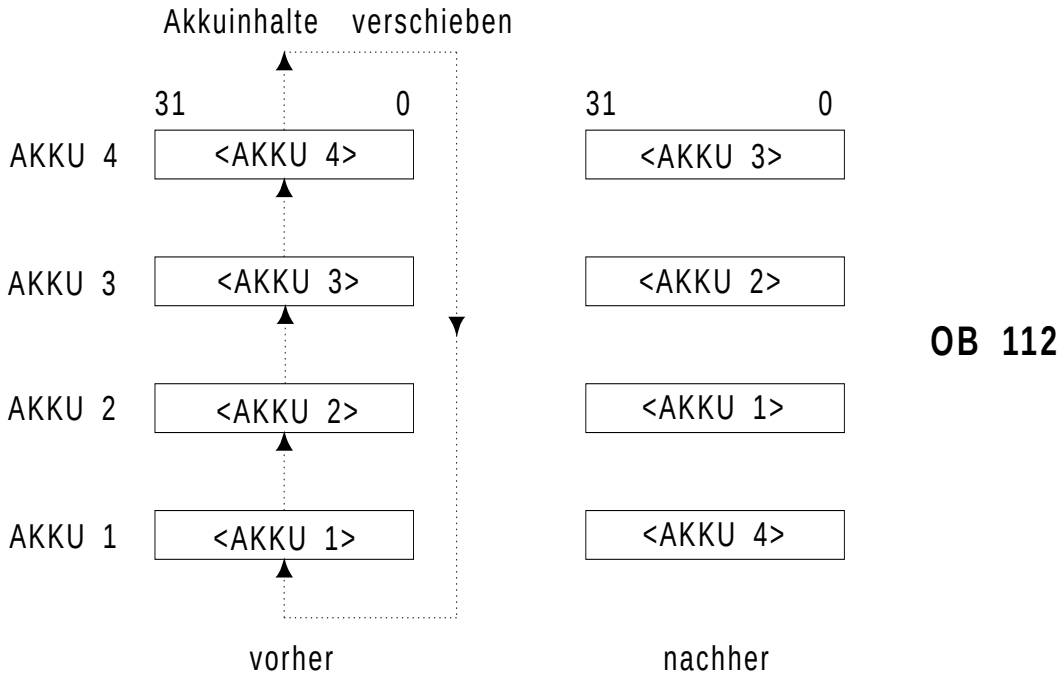


Bild 6-1 Wirkung der Funktion "Roll-Up"

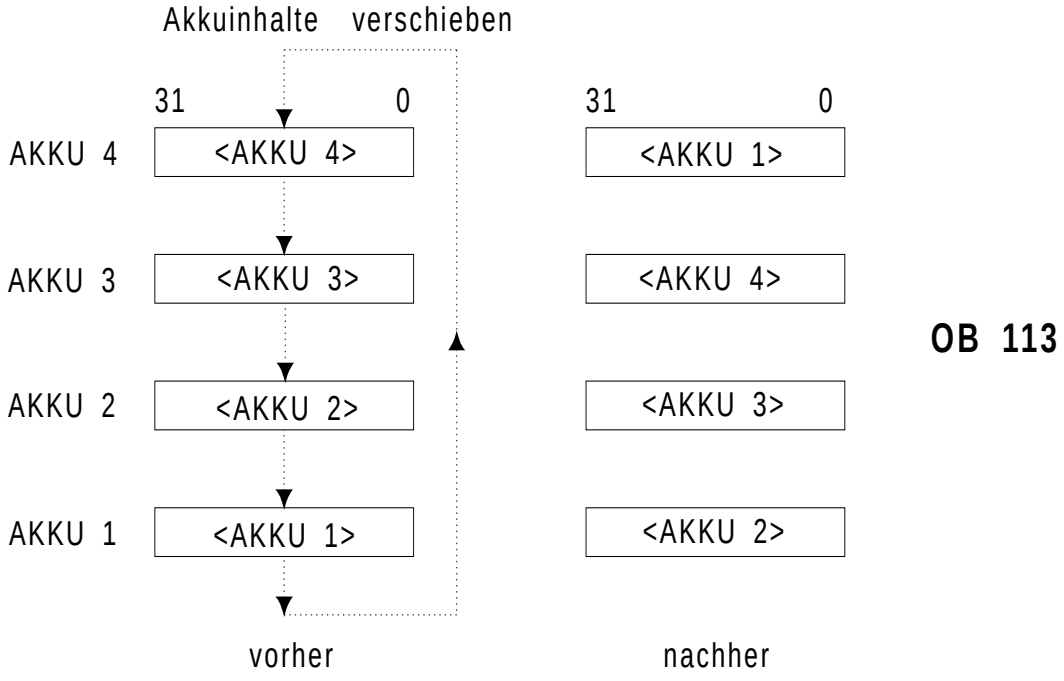


Bild 6-2 Wirkung der Funktion "Roll-Down"

6.5 OB 120: "Alarme gemeinsam sperren" ein-/ausschalten

Ein STEP-5-Programm kann an Baustein- oder Befehlsgrenzen von Programmen höherer Priorität unterbrochen werden. Zu diesen höherprioren Programmbearbeitungsebenen gehören die Prozeß- und alle Weckalarme (zyklische Weckalarme, uhrzeitgesteuerter Weckalarm und Verzögerungsalarm). Die Laufzeit des unterbrochenen Programms verlängert sich dabei jeweils um die Laufzeit der eingeschachtelten Programme.

Mit Hilfe des OB 120 können Sie das Einschachteln von höherprioren Programmbearbeitungsebenen an einer oder an mehreren aufeinanderfolgenden Baustein- oder Befehlsgrenzen (je nach Einstellung im DX 0) verhindern.

Funktion

Der OB 120 hat Auswirkungen auf die Reaktion auf Alarme:

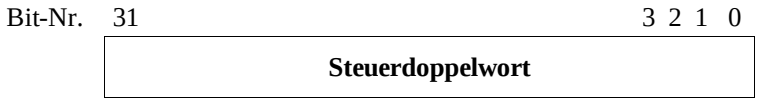
"Alarme sperren" **einschalten** heißt, es werden ab sofort keine Alarme mehr registriert und diejenigen Alarme, die bereits registriert worden sind (die z.B. auf eine Bausteingrenze "warten"), werden gelöscht. Nur falls der OB 2 (Prozeßalarm) oder ein OB für die Weckalarmbearbeitung bereits begonnen wurde, wird dieser vollständig bearbeitet.

"Alarme sperren" **ausschalten** heißt, es werden ab sofort alle auftretenden Alarme wieder registriert, an der nächsten Baustein- oder Befehlsgrenze eingeschachtelt und bearbeitet.

Parameter

1. Steuerdoppelwort

Der OB 120 vermerkt die zu sperrenden Alarme in einem systeminternen Steuerdoppelwort.



Die Bits des Steuerdoppelwortes haben folgende Bedeutung:

Bit-Nr. im Steuerdoppelwort	Funktion
0 = ’1’	**alle** zyklischen Weckalarme werden gesperrt
1 = ’1’	der uhrzeitgesteuerte Weckalarm wird gesperrt
2 = ’1’	**alle** Prozeßalarme werden gesperrt
3 = ’1’	der Verzögerungsalarm wird gesperrt
4 bis 31	reserviert; diese Bits müssen ’0’ sein!

Solange ein Bit auf ’1’ gesetzt ist, ist der betreffende Alarm gesperrt.

2. Akkus

2a) AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1, 2 oder 3 mit:

1: Der Inhalt von AKKU 1 wird in das Steuerwort geladen.

2: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**1**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

3: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**0**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

2b) AKKU 1

neues Steuerwort oder Maske, abhängig von der gewünschten Funktion

Ergebnis

Der Aufruf des OB 120 bringt folgende Ergebnisse:

Funkt.-Nr. in AKKU-2-L	Inhalt von AKKU 1	
	vorher	**nachher**
1	Steuerwort	Steuerwort
2	Maske	neues Steuerwort
3	Maske	neues Steuerwort

Fehlerfälle

- unzulässige Funktions-Nr. in AKKU-2-L
- eines der reservierten Bits in AKKU 1 (Nr. 4 bis 31) ist gleich ’1’

Im Fehlerfall wird der **OB 31** (Sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen wird in AKKU-1-L die Fehlerkennung **1A47H** hinterlegt.

Hinweise

- Der Zustand des Steuerworts läßt sich durch folgende Programmsequenz abfragen:

  1. Funktions-Nr. 2 oder 3 in den AKKU-2-L laden
  2. Wert ’0’ in den AKKU 1 laden
  3. Sonderfunktions-OB 120 aufrufen
  4. AKKU 1 auslesen

- Der Zustand der Alarmverarbeitung kann auch durch Auslesen der Systemdatenwortes BS 131 ermittelt werden.

  - BS 131 Anzeigenwort "Alarme gemeinsam sperren"

- Zum Sperren und Freigeben des Prozeßalarms können statt des OB 120 die Befehle AS und AF verwendet werden:

```
AS entspricht   :L    KB 2
                :L    KM  00000000 00000100
                :SPA  OB 120

AF entspricht   :L    KB 3
                :L    KM  00000000 00000100
                :SPA  OB 120
```

6.6 OB 121: "Weckalarme einzeln sperren" ein-/ausschalten

Mit Hilfe des OB 121 können Sie das Einschachteln von bestimmten Weckalarm-OBs (Weckalarme mit **festem Zeitraster**) an einer oder an mehreren aufeinanderfolgenden Baustein- oder Befehlsgrenzen verhindern.
Beispielsweise können Sie für einen bestimmten Programmteil festlegen, daß er nicht unterbrochen werden kann durch einen OB 18 (5 s) und einen OB 17 (2 s). Dagegen werden alle übrigen programmierten Weckalarme wie üblich bearbeitet

Funktion

Der OB 121 hat Auswirkungen auf die Reaktion auf zyklische Weckalarme:

"Weckalarme einzeln sperren" **einschalten** heißt, es werden ab sofort keine der angegebenen zyklischen Weckalarme mehr registriert und diejenigen Alarme, die bereits registriert worden sind (die z. B. auf eine Bausteingrenze "warten"), werden gelöscht. Nur falls ein Weckalarm-OB (zur Bearbeitung eines Weckalarms mit festem Zeitraster) bereits begonnen wurde, wird dieser vollständig bearbeitet.

"Weckalarme einzeln sperren" **ausschalten** heißt, es werden ab sofort alle auftretenden zyklischenWeckalarme wieder registriert, an der nächsten Baustein- oder Befehlsgrenze (je nach Einstellung im DX 0) eingeschachtelt und bearbeitet.

Parameter

1. Steuerwort

Der OB 121 vermerkt die zu sperrenden Weckalarme in einem systeminternen Steuerwort:

Die Bits des Steuerwortes haben folgende Bedeutung:

Bit-Nr.	Alarm
0 bis 2	reserviert, diese Bits müssen ’0’ sein!
3 = ’1’ 4 = ’1’ 5 = ’1’ 6 = ’1’ 7 = ’1’ 8 = ’1’ 9 = ’1’ 10 = ’1’ 11 = ’1’	zyklische Weckalarm mit festem Zeitraster: 10 ms (OB 10) 20 ms (OB 11) 50 ms (OB 12) 100 ms (OB 13) 200 ms (OB 14) 500 ms (OB 15) 1 s (OB 16) 2 s (OB 17) 5 s (OB 18)
12 bis 15	reserviert; diese Bits müssen ’0’ sein!

2. Akkus

2a) AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1, 2 oder 3 mit:

1: Der Inhalt von AKKU 1 wird in das Steuerwort geladen.

2: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**1**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

3: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**0**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

2b) AKKU 1

neues Steuerwort oder Maske, abhängig von der gewünschten Funktion

Fehlerfälle

- unzulässige Funktions-Nr. in AKKU-2-L
- eines der reservierten Bits in AKKU 1 (Nr. 4 bis 31) ist gleich ’1’

Im Fehlerfall wird der **OB 31** (Sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen wird in AKKU-1-L die Fehlerkennung **1A4AH** hinterlegt.

Hinweise

- Der Zustand des Steuerworts läßt sich durch folgende Programmsequenz abfragen:

  1. Funktions-Nr. 2 oder 3 in den AKKU-2-L laden
  2. Wert ’0’ in den AKKU 1 laden
  3. Sonderfunktions-OB 121 aufrufen
  4. AKKU 1 auslesen

- Der Zustand der Alarmverarbeitung kann auch durch Auslesen der Systemdatenwortes BS 135 ermittelt werden.

  - BS 135 Anzeigenwort "Weckalarme einzeln sperren"

6.7 OB 122: "Alarme gemeinsam verzögern" ein-/ausschalten

Ein STEP-5-Programm kann an Baustein- oder Befehlsgrenzen von Programmen höherer Priorität unterbrochen werden. Zu diesen höherprioren Programmbearbeitungsebenen gehören die Prozeß- und alle Weckalarme (zyklische Weckalarme, uhrzeitgesteuerter Weckalarm und Verzögerungsalarm). Die Laufzeit des unterbrochenen Programms verlängert sich dabei jeweils um die Laufzeit der eingeschachtelten Programme.

Mit Hilfe des OB 122 können Sie das Einschachteln von höherprioren Programmbearbeitungsebenen an einer oder an mehreren aufeinanderfolgenden Baustein- oder Befehlsgrenzen (je nach Einstellung im DX 0) verhindern.

Funktion

Der OB 122 hat Auswirkungen auf die Reaktion auf Alarme:

"Alarme verzögern" **einschalten** heißt, es werden weiterhin alle auftretenden Alarme registriert und bereits anstehende Alarme bleiben registriert. Eine Bearbeitung der registrierten Alarme findet jedoch zunächst nicht statt. Vorübergehend werden alle Befehls- bzw. Bausteingrenzen für die Bearbeitung von Alarmen unwirksam gemacht. Nur falls der OB 2 (Prozeßalarm) oder ein OB für die Weckalarmbearbeitung bereits begonnen wurde, wird dieser vollständig bearbeitet.

"Alarme verzögern" **ausschalten** heißt, es werden alle registrierten Alarme an der nächsten Baustein- oder Befehlsgrenze eingeschachtelt und bearbeitet.

> **Hinweis**
> Wird innerhalb der "Alarme verzögern"-Phase ein bestimmter Weckalarm-OB zum zweiten Mal aufgerufen, kommt es zum Weckfehler.

Parameter

1. Steuerdoppelwort

Der OB 122 vermerkt die zu verzögernden Alarme in einem systeminternen Steuerdoppelwort.

Bit-Nr.	31 ... 3 2 1 0
	Steuerdoppelwort

Die Bits des Steuerdoppelwortes haben folgende Bedeutung:

Bit-Nr. im Steuerdoppelwort	Funktion
0 = ’1’	**alle** zyklischen Weckalarme werden verzögert
1 = ’1’	der uhrzeitgesteuerte Weckalarm wird verzögert
2 = ’1’	**alle** Prozeßalarme werden verzögert
3 = ’1’	der Verzögerungsalarm wird verzögert
4 bis 31	reserviert; diese Bits müssen ’0’ sein!

Solange ein Bit auf ’1’ gesetzt ist, ist der betreffende Alarm verzögert.

2. Akkus

2a) AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1, 2 oder 3 mit :

1: Der Inhalt von AKKU 1 wird in das Steuerwort geladen.

2: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**1**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

3: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**0**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

2b) AKKU 1

neues Steuerwort oder Maske, abhängig von der gewünschten Funktion

Ergebnis

Der Aufruf des OB 122 bringt folgende Ergebnisse:

Funkt.-Nr. in AKKU-2-L	Inhalt von AKKU 1	
	vorher	**nachher**
1	Steuerwort	Steuerwort
2	Maske	neues Steuerwort
3	Maske	neues Steuerwort

Fehlerfälle

- unzulässige Funktions-Nr. in AKKU-2-L
- eines der reservierten Bits in AKKU 1 (Nr. 4 bis 31) ist gleich ’1’

Im Fehlerfall wird der **OB 31** (Sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen wird in AKKU-1-L die Fehlerkennung **1A48H** hinterlegt.

Hinweise

- Der Zustand des Steuerworts läßt sich durch folgende Programmsequenz abfragen:

  1. Funktions-Nr. 2 oder 3 in den AKKU-2-L laden
  2. Wert ’0’ in den AKKU 1 laden
  3. Sonderfunktions-OB 122 aufrufen
  4. AKKU 1 auslesen

- Der Zustand der Alarmverarbeitung kann auch durch Auslesen der Systemdatenwortes BS 132 ermittelt werden.

  - BS 132 Anzeigenwort "Alarme gemeinsam verzögern"

6.8 OB 123: "Weckalarme einzeln verzögern" ein-/ausschalten

Mit Hilfe des OB 123 können Sie das Einschachteln von bestimmten Weckalarm-OB (Weckalarme mit **festem Zeitraster**) an einer oder an mehreren aufeinanderfolgenden Baustein- oder Befehlsgrenzen verhindern.

Funktion

Der OB 123 hat Auswirkungen auf die Reaktion auf Weckalarme:

"Weckalarme einzeln verzögern" **einschalten** heißt, es werden weiterhin alle auftretenden Alarme registriert und bereits anstehende Alarme bleiben registriert. Eine Bearbeitung der registrierten Alarme findet jedoch zunächst nicht statt. Vorübergehend werden alle Befehls- bzw. Bausteingrenzen für die Bearbeitung von Alarmen unwirksam gemacht. Nur falls ein Weckalarm-OB (zur Bearbeitung eines Weckalarms mit festem Zeitraster) bereits begonnen wurde, wird dieser vollständig bearbeitet.

"Weckalarme einzeln verzögern" **ausschalten** heißt, es werden ab sofort alle auftretenden zyklischenWeckalarme wieder registriert, an der nächsten Baustein- oder Befehlsgrenze (je nach Einstellung im DX 0) eingeschachtelt und bearbeitet.

Hinweis
Wird innerhalb der "Alarme verzögern"-Phase ein bestimmter Weckalarm-OB zum zweiten Mal aufgerufen, kommt es zum Weckfehler.

Parameter

1. Steuerwort

Der OB 123 vermerkt die zu sperrenden Weckalarme in einem systeminternen Steuerwort:

Bit-Nr.	15	0
	S t e u e r w o r t	

Die Bits des Steuerwortes haben folgende Bedeutung:

Bit-Nr.	Alarm
0 bis 2	reserviert, diese Bits müssen ’0’ sein!
3 = ’1’ 4 = ’1’ 5 = ’1’ 6 = ’1’ 7 = ’1’ 8 = ’1’ 9 = ’1’ 10 = ’1’ 11 = ’1’	zyklische Weckalarm mit festem Zeitraster: 10 ms (OB 10) 20 ms (OB 11) 50 ms (OB 12) 100 ms (OB 13) 200 ms (OB 14) 500 ms (OB 15) 1 s (OB 16) 2 s (OB 17) 5 s (OB 18)
12 bis 15	reserviert; diese Bits müssen ’0’ sein!

2. Akkus

2a) AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1, 2 oder 3 mit:

1: Der Inhalt von AKKU 1 wird in das Steuerwort geladen.

2: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**1**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

3: Alle in der Maske in AKKU 1 mit einer ’1’ gekennzeichneten Bits werden im Steuerwort auf ’**0**’ gesetzt. Das neue Steuerwort wird in den AKKU 1 geladen.

2b) AKKU 1

neues Steuerwort oder Maske, abhängig von der gewünschten Funktion

Fehlerfälle

- unzulässige Funktions-Nr. in AKKU-2-L
- eines der reservierten Bits in AKKU 1 (Nr. 4 bis 31) ist gleich ’1’

Im Fehlerfall wird der **OB 31** (Sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen wird in AKKU-1-L die Fehlerkennung **1A4BH** hinterlegt.

Hinweise

- Der Zustand des Steuerworts läßt sich durch folgende Programmsequenz abfragen:

  1. Funktions-Nr. 2 oder 3 in den AKKU-2-L laden
  2. Wert ’0’ in den AKKU 1 laden
  3. Sonderfunktions-OB 123 aufrufen
  4. AKKU 1 auslesen

- Der Zustand der Alarmverarbeitung kann auch durch Auslesen der Systemdatenwortes BS 137 ermittelt werden.

  - BS 137 Anzeigenwort "Weckalarme einzeln verzögern"

6.9 OB 150: Systemzeit stellen/lesen

Eigenschaften der Systemzeit

- Die Auflösung beträgt 10ms beim Lesen und 1s beim Setzen.
- Schaltjahre werden berücksichtigt.
- Stundendarstellung wahlweise 24 Stunden oder 12 Stunden "am" (Vormittagszeit) und "pm" (Nachmittagszeit).
- Angabe des Wochentags
- Ein- und Ausgaben erfolgen BCD-codiert.
- Die integrierte Hardwareuhr wird über die Batterie im AG-Rahmen gepuffert. Wurde sie gestellt, so hat sie auch nach einem Spannungsausfall und WIEDERANLAUF den aktuellen Wert.

Funktion

Mit Hilfe des OB 150 können Sie in Ihrem Anwenderprogramm Datum und Uhrzeit der CPU 928B stellen oder auslesen. Datum und Uhrzeit werden im folgenden als "Systemzeit" bezeichnet.

> **Hinweis**
> Bevor Sie die Systemzeit auslesen können, muß sie zunächst **gestellt** werden.

Parameter

1. Datenfeld für die Zeitparameter

Beim **Stellen** der Systemzeit übernimmt der OB 150 die Zeit aus einem Datenfeld, beim **Lesen** der Zeit überträgt der OB 150 die aktuellen Daten in das Datenfeld. Sie können dieses Datenfeld in einem **Datenbaustein** oder in einem der beiden **Merkerbereiche** (M- oder S-Merker) anlegen.

Das Datenfeld besteht aus vier Wörtern.

1a) Format des Datenfeldes beim **Stellen** der Hardwareuhr

Bit-Nr.	15			12	11			8	7			4	3			0
1. Wort	Sekunden								0							
2. Wort	Format		Stunden						Minuten							
3. Wort	Monatstag								Wochentag				0			
4. Wort	Jahr								Monat							

1b) Format des Datenfeldes beim **Lesen** der Hardwareuhr

Bit-Nr.	15			12	11			8	7			4	3			0
1. Wort	Sekunden								1/100 Sekunde							
2. Wort	Format	Stunden							Minuten							
3. Wort	Monatstag								Wochentag				0			
4. Wort	Jahr								Monat							

Die Zeitparameter haben folgende Bedeutung, zulässige Wertebereiche und Darstellung:

Parameter	zulässiger Wertebereich	Darstellung
Sekunden 1/100 Sekunden Minuten Stunden Wochentag Monatstag [1] Monat Jahr	00 bis59 00 bis 99 00 bis 59 00 bis 23 oder 01 bis 12, je nach Angabe zu "Format" 0 bis 6 für Mo bis So 01 bis 31 01 bis 12 00 bis 09	BCD-Format
Format	Formatangabe für das Stundenfeld mit folgender Bedeutung: Bit 15 = 0: 12-Stunden-Format (Wahl "am" oder "pm" in Bit 14) Bit 15 = 1: 24-Stunden-Format (Bit 14 = 0) Bit 14 = 0: "am" Bit 14 = 1: "pm"	–

[1] Der angegebene Wert wird nach Aufruf des OB 150 logisch auf richtiges Datum unter Berücksichtigung der Schaltjahre überprüft.

Datenfeld im Merkerbereich

Wenn Sie das Datenfeld in einem Merkerbereich (M- oder S-Merker)anlegen, müssen Sie folgende Zuordnung der Datenfeldwörter zu den Merkerbytes berücksichtigen. Dabei ist ’x’ der Parameter "Nr.des 1. Datenfeldwortes", den Sie beim Aufruf des OB 150 im AKKU-1-L hinterlegen müssen:

Bit-Nr.	15 8	7 0
1. Datenfeldwort	Merkerbyte x	Merkerbyte x+1
2. Datenfeldwort	Merkerbyte x+2	Merkerbyte x+3
3. Datenfeldwort	Merkerbyte x+4	Merkerbyte x+5
4. Datenfeldwort	Merkerbyte x+6	Merkerbyte x+7

2. Akkus

2a) AKKU-2-L

Der AKKU-2-L enthält Angaben zur gewünschten Funktion und zum verwendeten Datenfeld. Er muß folgenden Aufbau haben:

Bit-Nr.	15			12	11			8	7			4	3			0
	Funktions-Nr.				Adreßbereichs-Typ				Datenbaustein-Nr.							

Parameter im AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1 = Systemzeit stellen
2 = Zystemzeit lesen

Adreßbereichs-Typ,
zulässige Werte: 1 = DB-Datenbaustein
2 = DX-Datenbaustein
3 = M-Merkerbereich
4 = S-Merkerbereich

Datenbaustein-Nr.,
zulässige Werte: 3 bis 255 (nur bei Adreßbereichs-Typ ’1’ oder ’2’; bei Adreßbereichs-Typ ’3’ oder ’4’ irrelevant)

2b) AKKU-1-L

Nummer des 1. Datenfeldwortes,
mögliche Werte (in Abhängigkeit vom Adreßbeeichs-Typ:

DB, DX:	0 bis 2044
M-Merker:	0 bis 248 (= Nr. Merkerbyte ’x ’)
S-Merker	0 bis 1016 (= Nr. Merkerbyte ’x ’)

Ergebnis

Nach korrekter Bearbeitung des OB 150 sind die Anzeigenbits OR, ERAB und OS = 0. Alle anderen Anzeigenbits sowie AKKU 1 und AKKU 2 sind unverändert.

Fehlerfälle

Im Fehlerfall werden der **OB 19** oder **OB 31** aufgerufen. Sind der OB 19 oder OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen werden in AKKU 1 und AKKU 2 Fehlerkennungen hinterlegt (siehe nachfolgende Tabelle).

Tabelle 6-2 Fehlerkennungen des OB 150

AKKU-1-L	AKKU-2-L	deutung	aufger. OB
1A07H	–	Datenbaustein nicht geladen	OB 19
1A4CH	0001H	Funktions-Nr. = 0 oder > 2	OB 31
	0010H	Adreßbereich-Typ unzulässig	
	0101H	Datenbaustein-Nr. unzulässig	
	0102H	"Nummer des ersten Datenfeldwortes" unzulässig	
	0103H	Datenbausteinlänge im Bausteinkopf < 5 Wörter	
	0201H	Jahresangabe im Datenfeld unzulässig	
	0202H	Monatsangabe im Datenfeld unzulässig	
	0203H	Monatstagangabe im Datenfeld unzulässig	
	0204H	Wochentagangabe im Datenfeld unzulässig	
	0205H	Stundenangabe im Datenfeld unzulässig	
	0206H	Minutenangabe im Datenfeld unzulässig	
	0207H	Sekundenangabe im Datenfeld unzulässig	
	0208H	1/100 Sekunde im Datenfeld ungleich 0	
	0209H	Datenfeldwort 3 /Bit-Nr. 0 bis 3 ≠ 0	
	020AH	Stunden-Format ungleich Einstellung bei OB 151	

Hinweis
Wird bei der Funktion "Systemzeit stellen" eine fehlerhafte Parametrierung vorgenommen und es wurde vorher schon mindestens einmal die Uhrzeit korrekt gestellt, so führt dies zwar zu der geschilderten Übergabe der Fehlerkennungen; die voher eingestellte Systemzeit läuft jedoch weiter.

Beispiele

```
"Systemzeit stellen":

Die Systemzeit soll auf folgende Werte eingestellt werden:

   "Do, 24.10.1993, 11:30 Uhr 0 Sekunden, 24-Stunden-Darstellung"

Es wird angenommen, daß die Zeitparameter im Datenbaustein DB 10 ab Datenwort
DW 0 hinterlegt werden. Das Stellen der Systemzeit mit den so vorbereiteten
Werten soll sekundengenau durch ein Eingangsbit (Triggerbit, z. B. E 1.0 –
Taster in der Nähe des AGs) ausgelöst werden.

Programmieren Sie zunächst den Datenbaustein DB 10 mit den nachfolgend aufge-
führten Werten und laden ihn ins AG. Die STEP-5-Operationen zum Aufrufen des
OB 150 programmieren Sie im OB 1 (zyklische Bearbeitung).  Programmieren Sie
den OB 1 so, daß die Operationen zum Aufruf des OB 151 nur bei der kommenden
Flanke des Triggerbits durchlaufen werden:
```

Fortsetzung auf der nächsten Seite

"Systemzeit stellen" *(Fortsetzung)*

```
   DB 10
        0: KH =  0 0 0 0    linkes Byte = Sekunden (BCD), rechtes Byte = 0

        1: KH =  9 1 3 0    91 = Format (=80H) + Stunde (=11 BCD)
                            30 = Minuten (BCD)
        2: KH =  2 4 3 0    24 = Monatstag (BCD)
                            30 = Wochentag (3 = Donnerstag) + Bit 0 bis 3 = 0
        3: KH =  9 3 1 0    91 = Jahr (BCD)
                            10 = Monat (BCD)

Im OB 1 hinterlegte STEP-5-Operationen zum Aufrufen des OB 150:

        :             Signalflanke des Eingangs zum Stellen der Systemzeit ist
        :             eingetroffen:
   STELL:L    KH  1 1 0 A  Werte für AKKU-2-L:
        :          | | |--- DB-Nr. = 10
        :          | |----- Adreßbereichs-Typ = 1 fuer "Datenfeld im DB"
        :          |------- Funktions-Nr. = 1 fuer "Stellen"
        :L    KF  0        AKKU-1-L:
        :                  Nr. 1. Datenfeldwort = 0
        :SPA  OB 150       OB 150 aufrufen
        :
```

"Systemzeit auslesen":

```
Die aktuelle Systemzeit soll in den Datenbaustein DB 10 ab Datenwort DW 4
geschrieben werden. Dazu müssen Sie den OB 150 mit folgenden Parametern
aufrufen:

        :
        :L    KH  2 1 0 A  Werte fuer AKKU-2-L:
        :          | | |--- DB-Nr. = 10
        :          | |----- Adreßbereichs-Typ = 1 fuer "Datenfeld im DB"
        :          |------- Funktions-Nr. = 2 fuer "Lesen"
        :L    KF  4        AKKU-1-L:
        :                  Nr. 1. Datenfeldwort = 4
        :SPA  OB 150       OB 150 aufrufen
        :A    DB 10        DB 10 aufschlagen
        :                  DB 10 auswerten

Nach dem Aufruf des OB 150 ist die aktuelle Systemzeit in folgender Form im
Datenbaustein DB 10 hinterlegt ("Do, 24.10.93, 11:30 Uhr 20 Sekunden,
13 Hundertstel-Sekunden, 24-Stunden-Darstellung"):

   DW 4:   KH =  2 0 1 3    Sekunden = 20 (BCD)
                            1/100 Sekunden = 13 (BCD)
   DW 5:   KH =  9 1 3 0    Format = 24 Std. (Bit 14/15 = 01), Stunden = 11(BCD)
                            Minuten = 30 (BCD)
   DW 6:   KH =  2 4 3 0    Monatstag = 24 (BCD)
                            Wochentag = 3 = Donnerstag
   DW 7:   KH =  9 3 1 0    Jahr = 91 (BCD)
                            Monat = 10 (BCD)
```

6.10 OB 151: Zeit für uhrzeitgesteuerten Weckalarm stellen/lesen

Funktion

Durch Aufruf des OB 151 können Sie

- die CPU 928B dazu veranlassen, zu einer vorgegebenen Zeit den uhrzeitgesteuertern Weckalarm ("Zeitauftrag" – OB 9, siehe Abschnitt 4.5.2) zu aktivieren:
  - minütlich
  - stündlich
  - täglich
  - wöchentlich
  - monatlich
  - jährlich
  - einmalig,
- den momentanen Status eines Zeitauftrags auslesen,
- einen bereits erzeugten Zeitauftrag stornieren.

Der OB 151 kann in den Betriebszuständen ANLAUF und RUN aufgerufen werden. Ein erzeugter uhrzeitgesteuerter Weckalarm bleibt bei WIEDERANLAUF (automatisch oder manuell) erhalten. Bei NEUSTART wird ein vorhandener Zeitauftrag gelöscht.

Wenn Sie einen neuen Zeitauftrag erzeugen, wird ein laufender automatisch storniert. Es ist also immer nur **ein** uhrzeitgesteuerter Weckalarm aktiv.

Parameter

1. Datenfeld für Auftragsparameter

Beim **Erzeugen** bzw. **Stornieren** eines Zeitauftrags entnimmt der OB 151 die benötigten Auftragsparameter einem Datenfeld.
Beim **Auslesen** des aktuellen Zustands der Auftragsverwaltung überträgt der OB 151 die aktuellen Auftragsparameter in ein Datenfeld.

Sie können dieses Datenfeld in einem **Datenbaustein** oder in einem der beiden **Merkerbereiche** (M- oder S-Merker) anlegen.

Das Datenfeld besteht aus vier Wörtern und hat beim Erzeugen und Auslesen eines Zeitauftrags folgendes Format:

Bit-Nr.	15			12	11			8	7			4	3			0
1. Wort	Sekunden								0							
2. Wort	Format		Stunden						Minuten							
3. Wort	Monatstag								Wochentag				Auftragsart			
4. Wort	Jahr								Monat							

Die Parameter haben folgende Bedeutung, zulässige Wertebereiche und Darstellung:

Parameter	zulässiger Wertebereich	Darstellung
Auftragsart	0 bis 7 mit: 0: Auftrag stornieren bzw. kein Auftrag aktiv 1: minütlich 2: stündlich 3: täglich 4: wöchentlich 5: monatlich 6: jährlich 7: einmalig	BCD-Format
Sekunden 1/100 Sekunden Minuten Stunden Wochentag Monatstag [1] Monat Jahr	00 bis59 00 bis 99 00 bis 59 00 bis 23 oder 01 bis 12, je nach Angabe zu "Format" 0 bis 6 für Mo bis So 01 bis 31 01 bis 12 00 bis 09	BCD-Format
Format [2]	Formatangabe für das Stundenfeld mit folgender Bedeutung: Bit 15 = 0: 12-Stunden-Format (Wahl "am" oder "pm" in Bit 14) Bit 15 = 1: 24-Stunden-Format (Bit 14 = 0) Bit 14 = 0: "am" Bit 14 = 1: "pm"	–

1) Der angegebene Wert wird nach Aufruf des OB 150 logisch auf richtiges Datum unter Berücksichtigung der Schaltjahre überprüft.

2) Bedeutung "am und "pm": siehe OB 150 im vorigen Abschnitt: "Format" muß mit der Formatangabe beim Stellen der Systemzeit durch OB 150 übereinstimmen!

Datenfeld im Merkerbereich

Wenn Sie das Datenfeld in einem Merkerbereich anlegen, müssen Sie folgende Zuordnung der Datenfeldwörter zu den Merkerbytes berücksichtigen. Dabei ist ’x’ der Parameter "Nr.des 1. Datenfeldwortes", den Sie beim Aufruf des OB 151 im AKKU-1-L hinterlegen müssen:

Bit-Nr.	15 8	7 0
1. Datenfeldwort	Merkerbyte x	Merkerbyte x+1
2. Datenfeldwort	Merkerbyte x+2	Merkerbyte x+3
3. Datenfeldwort	Merkerbyte x+4	Merkerbyte x+5
4. Datenfeldwort	Merkerbyte x+6	Merkerbyte x+7

2. Akkus

2a) AKKU-2-L

Der AKKU-2-L enthält Angaben zur gewünschten Funktion und zum verwendeten Datenfeld. Er muß folgenden Aufbau haben:

Bit-Nr.	15			12	11			8	7			4	3			0
	Funktions-Nr.				Adreßbereichs-Typ				Datenbaustein-Nr.							

Parameter im AKKU-2-L

Funktions-Nr.,
zulässige Werte: 1 = Auftrag erzeugen
2 = Auftragsstatus lesen

Adreßbereichs-Typ,
zulässige Werte: 1 = DB-Datenbaustein
2 = DX-Datenbaustein
3 = M-Merkerbereich
4 = S-Merkerbereich

Datenbaustein-Nr.,
zulässige Werte: 3 bis 255 (nur bei Adreßbereichs-Typ ’1’ oder ’2’; bei Adreßbereichs-Typ ’3’ oder ’4’ irrelevant)

2b) AKKU-1-L

Nummer des 1. Datenfeldwortes,
mögliche Werte (in Abhängigkeit vom Adreßbeeichs-Typ:

DB, DX:	0 bis 2044
M-Merker:	0 bis 248 (= Nr. Merkerbyte ’x ’)
S-Merker	0 bis 1016 (= Nr. Merkerbyte ’x ’)

Hinweis
Es ist nicht sinnvoll, einen Zeitauftrag zyklisch zu erzeugen (z. B. durch einen absoluten Aufruf des OB 151 mit der Funktions-Nr. ’1’ im OB 1).

Ergebnis

Nach korrekter Bearbeitung des OB 150 sind die Anzeigenbits OR, ERAB und OS = 0. Alle anderen Anzeigenbits sowie AKKU 1 und AKKU 2 sind unverändert.

Hinweis
Wenn beim Auslesen des Zeitauftrags im Datenfeld die **Auftragsart '0'** und alle restlichen Parameter **'F'** oder **'FF'** (hexadezimal) sind, ist kein Zeitauftrag aktiv.
Dieser Zustand kann entstehen

a) wenn NEUSTART durchgeführt wurde, ohne einen Zeitauftrag zu erzeugen,

b) wenn ein einmaliger Zeitauftrag fällig war

oder

c) wenn Sie einen Auftrag storniert haben.

Fehlerfälle

Im Fehlerfall werden der **OB 19** oder **OB 31** aufgerufen. Sind der OB 19 oder OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen werden in AKKU 1 und AKKU 2 Fehlerkennungen hinterlegt (siehe nachfolgende Tabelle).

Tabelle 6-3 Fehlerkennungen des OB 151

AKKU-1-L	AKU-2-L	Bedeutung	aufger. OB
1A07H	–	Datenbaustein nicht geladen	OB 19
1A4DH	0001H	Funktions-Nr. = 0 oder > 2	OB 31
	0100H	Adreßbereich-Typ unzulässig	
	0101H	Datenbaustein-Nr. unzulässig	
	0102H	"Nummer des ersten Datenfeldwortes" unzulässig	
	0103H	Datenbausteinlänge im Bausteinkopf < 5 Wörter	
	0201H	Jahresangabe im Datenfeld unzulässig	
	0202H	Monatsangabe im Datenfeld unzulässig	
	0203H	Monatstagangabe im Datenfeld unzulässig	
	0204H	Wochentagangabe im Datenfeld unzulässig	
	0205H	Stundenangabe im Datenfeld unzulässig	
	0206H	Minutenangabe im Datenfeld unzulässig	
	0207H	Sekundenangabe im Datenfeld unzulässig	
	0208H	1/100 Sekunde im Datenfeld ungleich 0	
	0209H	Auftragart im Datenfeld > 7	
	020AH	Stunden-Format ungleich Einstellung bei OB 150	

Hinweis
Wird eine fehlerhafte Parametrierung vorgenommen **und** es wurde vorher schon ein gültiger Zeitauftrag erzeugt, so führt dies zwar zu der oben geschilderten Übergabe der Fehlerkennungen; **der vorher erzeugte Zeitauftrag bleibt jedoch weiter bestehen**.

Was Sie bei den Zeitparametern beachten müssen

Abhängig davon, wann ein uhrzeitgesteuerter Weckalarm (Zeitauftrag) ausgelöst werden soll, müssen Sie die einzelnen Zeitparameter in bestimmten Kombinationen vorgeben. Dabei müssen Sie – abhängig vom gewählten Zeitpunkt für den uhrzeitgesteuerten Weckalarm – bestimmte Parameter unbedingt angeben, während andere vom Systemprogramm nicht ausgewertet werden und daher entfallen können.

Nachfolgende Tabelle gibt Ihnen an, welche Zeitparameter bei welchem Zeitauftrag angegeben werden müssen (XXX = notwendige Angabe, --- = irrelevant).

Tabelle 6-4 Zuordnung "Zeitauftrag – Zeitparameter"

Weckzeit	Sekunden	Minuten	Stunden	Wochentag	Monatstag	Monat	Jahr
minütlich	XXX	---	---	---	---	---	---
stündlich	XXX	XXX	---	---	---	---	---
täglich	XXX	XXX	XXX	---	---	---	---
wöchentlich	XXX	XXX	XXX	XXX	---	---	---
monatlich	XXX	XXX	XXX	---	XXX	---	---
jährlich	XXX	XXX	XXX	---	XXX	XXX	---
einmalig	XXX	XXX	XXX	---	XXX	XXX	XXX

Besonderheiten

- Wenn bei Auftragsart "jährlich" (= 6) der "29. Februar" als Monatstag und Monat gewählt wird, so resultiert daraus , daß der OB 9 nur in jedem Schaltjahr aufgerufen wird.
- Wenn bei Auftragsart "monatlich" (= 5) als Monatstag der Wert "29", "30" oder "31" gewählt wird, so wird der OB 9 nur in den Monaten aufgerufen, bei denen dieser Tag vorhanden ist.

Beispiele

```
Verschiedene Zeitaufträge (im 24-Stunden-Anzeigeformat):

1. "Auftrag jede Minute beim Sekundenwert 29"
    (12:44:29, 12:45:29 usw.):

   Sie müssen angeben:      Auftragsart =     1  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    29


2. "Auftrag stündlich jeweils um xx:14:15 Uhr":

   Sie müssen angeben:      Auftragsart =     2  (Funktions-Nr. in AKKU-2-L =  1)
                            Sekunden    =    15
                            Minuten     =    14

                                          Fortsetzung auf der nächsten Seite
```

Verschiedene Zeitaufträge (im 24-Stunden-Anzeigeformat), *1.Fortsetzung* :

```
3. "Auftrag täglich um 5:32:47 Uhr":

   Sie müssen angeben:      Auftragsart =     3  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    47
                            Minuten     =    32
                            Stunden     =    05


4. "Auftrag wöchentlich, dienstags um10:50:00 Uhr":

   Sie müssen angeben:      Auftragsart =     4  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    00
                            Minuten     =    50
                            Stunden     =    10
                            Wochentag   =    01


5. "Auftrag monatlich, jeweils am 14. um 7:30:15 Uhr":

   Sie müssen angeben:      Auftragsart =     5  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    15
                            Minuten     =    30
                            Stunden     =    07
                            Monatstag   =    14


6. "Auftrag jährlich, jeweils am 1.5. um 00:01:45 Uhr":

   Sie müssen angeben:      Auftragsart =     6  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    45
                            Minuten     =    01
                            Stunden     =    00
                            Monatstag   =    01
                            Monat       =    05


7. "Auftrag einmalig am 31.12.1999 um 23:55:00 Uhr":

   Sie müssen angeben:      Auftragsart =     7  (Funktions-Nr. in AKKU-2-L = 1)
                            Sekunden    =    00
                            Minuten     =    55
                            Stunden     =    23
                            Monatstag   =    31
                            Monat       =    12
                            Jahr        =    99
```

Fortsetzung auf der nächsten Seite

Verschiedene Zeitaufträge (im 24-Stunden-Anzeigeformat), *2.Fortsetzung* :

```
8.                          "Auftrag stornieren":

   Sie müssen angeben:      Auftragsart =     0   (Funktions-Nr. in AKKU-2-L = 1)


9. "Zeitauftrag auslesen":

   Sie müssen angeben:      Funktions-Nr. in AKKU-2-L = 2

   Falls kein Auftrag aktiv ist, erhalten Sie als Ergebnis im Datenfeld:

                            Datenfeldwort 0:     FFFF H
                            Datenfeldwort 1:     FFFF H
                            Datenfeldwort 2:     FFF0 H
                            Datenfeldwort 3:     FFFF H
```

6.11 OB 152: Zyklusstatistik

In der CPU 928B können eine Reihe statistischer Daten zur Zyklusdauer erfaßt werden (Zyklusstatistik). Mit Hilfe des OB 152 können Sie die Zyklusstatistik initialisieren, die Statistikdaten auslesen und das Erfassen der Statistikdaten ein- und ausschalten.

Übersicht

Die statistischen Daten umfassen

- die Dauer des vorangegangenen Zyklus,
- die seit der letzten Zyklusgrenze im aktuell bearbeiteten Zyklus verstrichene Zykluszeit,
- die minimale und die maximale Zykluszeit seit der letzten Initialisierung der Zyklusstatistik,
- die Anzahl der Zyklen, die seit der letzten Initialisierung der Zyklusstatistik in die Statistik aufgenommen wurden,
- den Mittelwert der Zykluszeit: Es werden maximal die letzten 256 in der Statistik erfassten Zyklen zur Berechnung des Mittelwerts verwendet.

Hinweis
In der Zyklusstatistik werden nur "normale" Zyklen erfaßt. Wurde das Erfassen der Dauer des aktuellen Zyklus z. B. durch Nachtriggern oder Neustarten der Zyklusüberwachungszeit verfälscht, so werden die Daten dieses Zyklus **nicht** in die Statistik übernommen. Dadurch wirken sich "Ausreißer" nicht auf die Statistik aus.
Dies bewirkt aber auch, daß z. B. bei wiederholtem Nachtriggern der Zyklusüberwachungszeit nur wenige oder überhaupt keine Daten für die Statistik erfaßt werden (beachten Sie hierzu die Hinweise am Ende von Abschnitt 6.8 "Verfälschung der statistischen Daten").

Ein-/Ausschalten der Statistik

Nach NEUSTART (automatisch oder manuell) ist die Statistik-Funktion **immer** ausgeschaltet und die statistischen Daten sind **gelöscht** (die Zyklusstatistik ist initialisiert). Ein WIEDERANLAUF (automatisch oder manuell) verändert weder den Zustand der Statistik noch die statistischen Daten.

Sie können die Statistikfunktion im Betriebszustand ANLAUF oder RUN mit Hilfe des OB 152 einschalten.

Wurde die Statistik-Funktion mit dem OB 152 eingeschaltet, so werden die statistischen Daten an jeder Zyklusgrenze aktualisiert und können über Aufruf des OB 152 ausgelesen werden.

Wenn Sie die Statistikfunktion nicht mehr benötigen, so können Sie sie im Betriebszustand ANLAUF oder RUN über den OB 152 wieder ausschalten. Dadurch wird die Belastung der Zykluszeit, die durch das Aktualisieren der Zyklusdaten an jeder Zyklusgrenze entsteht, klein gehalten.

Initialisieren können Sie die Zyklusstatistik ebenfalls über den OB 152 im ANLAUF oder im RUN. Sinnvoll wäre es z. B. nach Auswertung der statistischen Daten (u. U. in Abhängigkeit vom Wert des Zyklus-Zählers) die Zyklusstatistik zu initialisieren.

Statistik-Daten

Die Statistik-Daten werden als einzelne Größen entweder direkt über den OB 152 ausgelesen oder beim Aufruf des OB 152 berechnet. Sie werden vom OB 152 in AKKU-1-L bzw. AKKU-2-L übergeben.

Folgende Statistik-Größen können Sie durch Aufruf des OB 152 ermitteln:

Tabelle 6-5 Größen der Zyklusstatistik – OB 152

Statistik-Größe	Bedeutung	Format	Einheit	Wertebereich
AKTZYK	bereits abgelaufene Zeit des aktuellen Zyklus	Festpunktzahl	Millisekunden	0 bis 13 000
LASTZYK	Dauer des letzten abgelaufenen Zyklus	Festpunktzahl	Millisekunden	0 bis 13 000
MINZYK	Dauer des kürzesten Zyklus seit der letzten Initialisierung der Zyklusstatistik	Festpunktzahl	Millisekunden	0 bis 13 000
MAXZYK	Dauer des längsten Zyklus seit der letzten Initialisierung der Zyklusstatistik	Festpunktzahl	Millisekunden	0 bis 13 000
MITTEL-WERT	Mittelwert der Zykluszeiten der zuletzt abgelaufenen - maximal 256 - Zyklen [1)]	Festpunktzahl	Millisekunden	0 bis 13 000
ZYKLUS-ZÄHLER	Anzahl der seit der letzten Initialisierung der Zyklusstatistik in die Statistik aufgenommenen Zyklen.	Hexadezimal-zahl	Anzahl der Zyklen	0 bis 0FFFFH

1) siehe "Berechnung des Miotelwertes"

Berechnung des Mittelwertes

Der Mittelwert wird durch den OB 152 nach folgendem Algorithmus berechnet:

In einem systeminternen Umlaufpuffer wird bei jedem Aktualisieren der statistischen Daten der Wert von LASTZYK gespeichert. Dieser Puffer kann maximal 256 Werte aufnehmen. Ist der Puffer gefüllt, so fällt der jeweils älteste Wert von LASTZYK heraus, und der neueste Wert wird abgespeichert. Zusätzlich wird bei jeder Aktualisierung der Daten der Summenwert von den im Puffer enthaltenen LASTZYK-Werten derart gebildet, daß immer die **neuesten LASTZYK-Werte** (maximal 256) darin enthalten sind.

Bei Aufruf des OB 152 wird der Mittelwert gebildet durch Division des Summenwertes durch die Anzahl LASTZYK-Werte, die im Puffer abgelegt sind. Praktisch bedeutet dies, daß der Mittelwert fast immer aus den LASTZYK-Werten der **letzten 256 Zyklen** gebildet wird.

Funktionen

Bei Aufruf des OB 152 können Sie über eine Funktions-Nr. folgende Einzelfunktionen aktivieren:

Tabelle 6-6 Funktionen des OB 152

Funktions-Nr.	Funktion
0	Zyklusstatistik ausschalten
1	AKTZYK / LASTZYK lesen
2	MINZYK / MAXZYK lesen
3	MITTELWERT / ZYKLUS-ZÄHLER lesen
8	Zyklusstatistik initialisieren
15	Zyklusstatistik einschalten

Parameter

AKKU-1-L

Der AKKU-1-L enthält die Funktions-Nr.; er muß folgenden Aufbau haben:

Bit-Nr.	15 ... 4	3 ... 0
	0	Funktions-Nr.

Funktions-Nr.
zulässige Werte: siehe Tabelle 6-4

Bit-Nr. 4 bis 15 müssen ’0’ sein!

Ergebnis

Nach Aufruf des OB 152 sind die Anzeigen OS, OR und $\overline{\text{ERAB}}$ = ’0’, das VKE ist bis auf die nachfolgend aufgeführten Fälle ebenfalls = ’0’. Außerdem werden in AKKU-1-L und AKKU-2-L die bei einigen Funktionen angeforderten Statistik-Größen übergeben (siehe nachfolgende Tabelle).

Tabelle 6-7 Funktionsergebnisse des OB 152

Funktion	**Funktionsergebnisse**		
	AKKU-1-L	**AKKU-2-L**	**Bedeutung von "VKE = 1"**
Zyklusstatistik ausschalten	unverändert		–
AKTZYK / LASTZYK lesen	AKTZYK	LASTZYK	AKTZYK ist unkorrekt, Daten des aktuellen Zyklus werden nicht in die Statistik übernommen. [1]
MINZYK / MAXZYK lesen	MINZYK	MAXZYK	–
MITTELWERT / ZYKLUS-ZÄHLER lesen	MITTEL-WERT	ZYKLUS-ZÄHLER	Überlauf ZYKLUS-ZÄHLER [2]
Zyklusstatistik initialisieren	unverändert		–
Zyklusstatistik einschalten	unverändert		–

[1] Durch Neustarten/Nachtriggern der Zyklusüberwachungszeit, Zyklusfehler oder WIEDERANLAUF

[2] Ist beim Auslesen des Zykluszählers VKE = ’1’ gesetzt, so wird gleichzeitig mit der Anzeigenübergabe ein systeminterner Merker für Zyklusüberlauf gelöscht. Dieser Merker wird danach erst wieder gesetzt, wenn der Zykluszähler erneut übergelaufen ist.

Fehlerfall

Der Fehlerfall tritt ein, wenn im AKKU-1-L eine falsche Funktions-Nr. übergeben wird (erlaubt sind nur die Nummern 0 bis 3, 8 und 15).

Es wird der **OB 31** (sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.

In beiden Fällen wird in AKKU-1-L die Fehlerennung **1A4EH** und in AKKU-2-L Fehlerkennung **0001H** hinterlegt.

Besonderheiten

Nachfolgend erhalten Sie Hinweise auf einige Besonderheiten des OB 152 bei NEUSTART, nach einem ANLAUF oder bei Eintreffen bestimmter Ereignisse hin, die Sie bei seiner Benutzung beachten sollten.

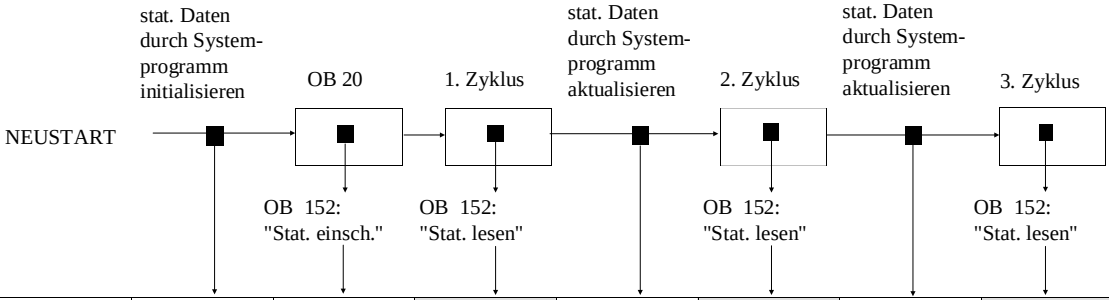
Verhalten nach NEUSTART

Bei NEUSTART werden die statistischen Daten initialisiert. Ein Aufruf des OB 152 im 1. Zyklus nach NEUSTART liefert die Initialisierungsdaten zurück.
Nachfolgende Tabelle zeigt Ihnen, wie die statistischen Daten vom Systemprogramm

- nach NEUSTART initialisiert

  und

- in den ersten drei Zyklen aktualisiert werden.



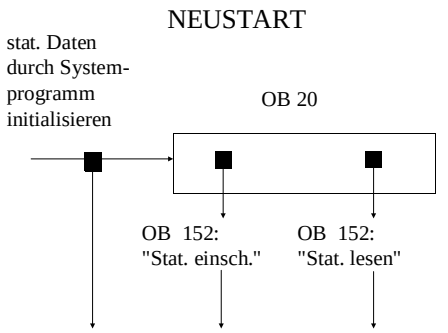
AKTZYK [1]	---	---	AKTZYK (1.)	---	AKTZYK (2.)	---	AKTZYK (3.)
LASTZYK	0	0	0	Zykl.-Zeit (1.)	Zykl.-Zeit (1.)	Zykl.-Zeit (2.)	Zykl.-Zeit (2.)
MINZYK	13 000	13 000	13 000	Zykl.-Zeit (1.)	Zykl.-Zeit (1.)	min. Z.-Z.	min. Z.-Z.
MAXZYK	0	0	0	Zykl.-Zeit (1.)	Zykl.-Zeit (1.)	max. Z.-Z.	max. Z.-Z.
MITTELW.	0	0	0	Zykl.-Zeit (1.)	Zykl.-Zeit (1.)	mittl. Z.-Z.	mittl. Z.-Z.
ZYKL.-Z.	0	0	0	1	1	2	2

[1] Der Wert für AKTZYK wird jeweils beim Auslesen über OB 152 dem Zyklusüberwachungstimer entnommen. Daher steht er auch im 1. Zyklus schon zur Verfügung

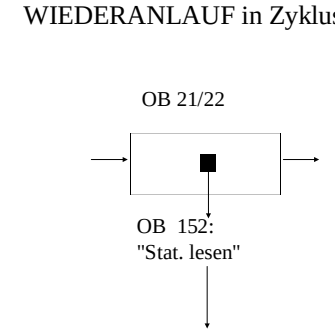
Beim Initialisieren der statistischen Daten werden außer den in der Tabelle aufgeführten Vorbesetzungen der Daten systemintern der Umlaufpuffer für die Mittelwertbildung gelöscht sowie ein interner Merker für Zykluszählerüberlauf rückgesetzt.

Aufruf des OB 152 in einem Anlauf-OB

Je nach Anlaufart liefert der Aufruf des OB 152 zum Lesen der statistischen Daten in AKKU-1-L und AKKU-2-L folgende Werte (grau unterlegte Spalten):



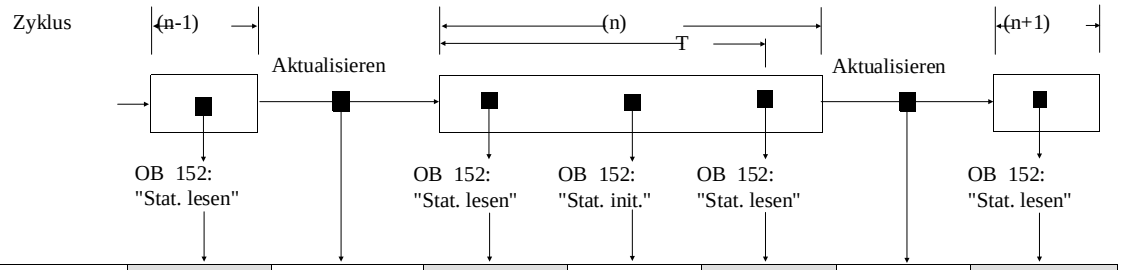
AKTZYK	---	---	0
LASTZYK	0	0	0
MINZYK	13 000	13 000	13 000
MAXZYK	0	0	0
MITTELW.	0	0	0
ZYKL.-Z.	0	0	0



AKTZYK	0
LASTZYK	Zykl.-Zeit (n-1)
MINZYK	incl. Zyk. (n-1)
MAXZYK	incl. Zyk. (n-1)
MITTELW.	incl. Zyk. (n-1)
ZYKL.-Z.	n-1

Initialisieren der statistischen Daten durch Aufruf des OB 152

Nachfolgende Tabelle zeigt Ihnen, wie bei einer Initialiserung der statistischen Daten diese durch Aufruf des OB 152 im ZYKLUS verändert werden. Die grau unterlegten Spalten enthalten die beim Lesen der statistischen Daten übergebenen Werte.



AKTZYK	AKTZYK (n-1)	---	AKTZYK (n)	---	T	---	AKTZYK (n+1)
LASTZYK	Zyk.-Zeit (n-2)	Zyk.-Zeit (n-1)	Zyk.-Zeit (n-1)	0	0	nein	0
MINZYK	incl. Zyk. (n-2)	incl. Zyk. (n-1)	incl. Zyk. (n-1)	13 000	13 000	nein	13 000
MAXZYK	incl. Zyk. (n-2)	incl. Zyk. (n-1)	incl. Zyk. (n-1)	0	0	nein	0
MITTELW.	incl. Zyk. (n-2)	incl. Zyk. (n-1)	incl. Zyk. (n-1)	0	0	nein	0
ZYKL.-Z.	n-2	n-1	n-1	0	0	nein	0

Beim Initialisieren der statistischen Daten werden außer den in der Tabelle aufgeführten Vorbesetzungen der Daten systemintern der Umlaufpuffer für die Mittelwertbildung gelöscht sowie ein interner Merker für Zykluszählerüberlauf rückgesetzt.

Nach dem Initialisieren der statistischen Daten durch Aufruf des OB 152 werden die Daten durch das Systemprogramm erst am **Ende** des 1. Zyklus, der **nach** der Initialisierung folgt, aktualisiert!

Aufruf des OB 152 bei ausgeschalteter Zyklusstatistik

Wird die Zyklusstatistik durch Aufruf des OB 152 ausgeschaltet, so bleiben die statistischen Daten **der letzten Aktualisierung** erhalten. Wird danach der OB 152 zum Lesen von Statistik-Daten aufgerufen, so liefert er entsprechend die Daten der letzten Aktualisierung vor dem Ausschalten.

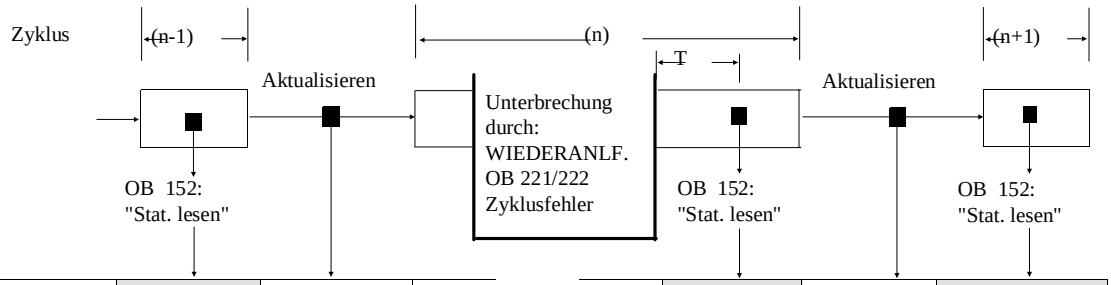
Werden die statitistischen Daten nach einem NEUSTART gelesen, ohne daß die Zyklusstatistik durch Aufruf des OB 152 eingeschaltet wurde, so liefert der OB 152 die Initialisierungsdaten.

Verfälschung der statistischen Daten

Durch das Auftreten bestimmter Ereignisse wird die Erfassung der Zyklusdauer für den aktuellen Zyklus gestört und führt dann zu falschen Werten. Daher werden in solchen Fällen die statistischen Daten für den betroffenen Zyklus nicht aktualisiert.

Zu diesen Ereignissen gehören:

- WIEDERANLAUF
- Neustarten der Zyklusüberwachungszeit durch Aufruf des OB 221
- Nachtriggern der Zyklusüberwachungszeit durch Aufruf des OB 222
- Zyklusfehler



AKTZYK	AKTZYK (n-1)	---			1)	---	AKTZYK (n+1)
LASTZYK	Zyk.-Zeit (n-2)	Zyk.-Zeit (n-1)			Zyk.-Zeit (n-1)	nein	Zykl.-Zeit (n-1)
MINZYK	incl. Zyk. (n-2)	incl. Zyk. (n-1)			Zyk.-Zeit (n-1)	nein	incl. Zyk. (n-1)
MAXZYK	incl. Zyk. (n-2)	incl. Zyk. (n-1)			Zyk.-Zeit (n-1)	nein	incl. Zyk. (n-1)
MITTELW.	incl. Zyk. (n-2)	incl. Zyk. (n-1)			Zyk.-Zeit (n-1)	nein	incl. Zyk. (n-1)
ZYKL.-Z.	n-2	n-1			n-1	nein	n-1

1) Der Wert von AKTZYK entspricht der Zeit T, die seit dem Auftreten der "Störung" im aktuellen Zyklus verstrichen ist. Dies ist nicht die Dauer des gesamten Zyklus. Um diesen Fall zu kennzeichnen, wird zusätzlich zu den in AKKU-1-L und AKKU-2-L übergebenen Werten das VKE = '1' gesetzt.

6.12 OB 153: Zeit für Verzögerungsalarm stellen/lesen

Über den OB 153 können Sie sogenannte "Verzögerungsaufträge" an das Systemprogramm übergeben. Diese führen dazu, daß nach Ablauf einer vorgegebenen Verzögerungszeit ein "Verzögerungsalarm" bearbeitet wird (siehe OB 6, Abschnitt 4..5.3).

Funktion

Durch Aufruf des OB 153 können Sie

- eine Verzögerungszeit definieren und starten,
- eine aktivierte Verzögerungszeit stoppen (Verzögerungsauftrag stornieren),
- die aktuelle Restzeit einer aktivierten Verzögerungszeit lesen.

Ein Verzögerungsauftrag kann über den OB 153 in den Betriebszuständen ANLAUF und RUN **abgegeben** werden.

Lebensdauer eines Verzögerungsauftrags

Der durch einen Verzögerungsauftrag ausgelöste Verzögerungsalarm wird jedoch vom Systemprogramm **nur im RUN aktiviert** (Aufruf des OB 6).
Aufträge, die außerhalb des RUN fällig werden, werden vom Systemprogramm **ohne Meldung verworfen**.
Ein laufender (und noch nicht fälliger) Auftrag wird außerdem verworfen beim Übergang in den STOP sowie bei NETZ AUS.

Parameter

Akkus

a) AKKU-2-L

Verzögerungszeit in Millisekunden (max. 65535)

zulässige Werte: 0001H bis FFFFH

Der AKKU-2-L muß nur bei Aufruf des OB 153 nur für die Funktion "Verzögerungszeit definieren" (Funktions-Nr. 1) versorgt werden. Bei den übrigen Funktionen des OB 153 wird der Inhalt von AKKU-2-L nicht ausgewertet.

b) AKKU-1-L

Funktions-Nr.

zulässige Werte: 1 = Verzögerungszeit definieren und starten
2 = Verzögerungszeit stoppen (= Auftrag storn.)
3 = aktuelle Restzeit lesen

Hinweis
Ist beim Definieren einer Verzögerungszeit eine vorher definierte Verzögerungszeit noch nicht abgelaufen, so wird diese "vergessen" und die neue Verzögerungszeit gestartet.

Ergebnis

Nach korrekter Bearbeitung des OB 153 sind die Anzeigenbits OR, ERAB und OS = 0.

Bei Aufruf des OB 153 mit der Funktions-Nr. ’3’ enthält AKKU-1-L die Restzeit in Millisekunden.

Ist bei Aufruf des OB 153 mit der Funktions-Nr. ’2’ oder ’3’ kein Verzögerungsauftrag aktiv, so enthält AKKU-1-L den Wert ’0’.

Fehlerfälle

Es können die in der nachfolgenden Tabelle aufgeführten Fehlerfälle auftreten.
Es wird der **OB 31** (sonstige Laufzeitfehler) aufgerufen. Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.

In beiden Fällen werden in AKKU 1und AKKU-2-L Fehlerkennung hinterlegt (siehe nachfolgende Tabelle).

Tabelle 6-8 Fehlerkennungen des OB 153

AKKU-1-L	AKKU-2-L	Bedeutung
1A4FH	0001H 0002H	Funktions-Nr. = 0 oder > 3 Verzögerungszeit unzulässig

Beispiele

Verzögerungszeit definieren und starten:

```
Bei einem AUTOMATISCHEN WIEDERANLAUF soll nach 5 Sekunden einmalig eine
bestimmte STEP-5-Operationsfolge durchlaufen werden. Dazu wird im
Anlauforganisationsbaustein OB 22 die Verzögerungszeit definiert und gestartet.

Im OB 22 hinterlegte STEP-5-Operationen zum Aufrufen des OB 153:

        :
        :
        :L    KF  +5000     Wert fuer AKKU-2-L: 5000 ms
        :L    KF  +1        Wert fuer AKKU-1-L: Funktions-Nr. = 1 fuer
        :                   "Verzoegerungszeit definieren und starten"
        :SPA  OB 153        OB 153 aufrufen
        :
```

Verzögerungszeit stoppen (Auftrag stornieren):

STEP-5-Operationen zum Aufrufen des OB 153:

```
        :
        :
        :L    KF  +2        Wert fuer AKKU-1-L: Funktions-Nr. = 2 fuer
        :                   "Verzoegerungszeit stoppen"
        :SPA  OB 153        OB 153 aufrufen
        :
        :
```

Restzeit eines Verzögerungsauftrags auslesen):

STEP-5-Operationen zum Aufrufen des OB 153:

```
        :
        :
        :L    KF  +3        Wert fuer AKKU-1-L: Funktions-Nr. = 2 fuer
        :                   "Restzeit auslesen"
        :SPA  OB 153        OB 153 aufrufen
        :
        :                   AKKU-1-L enthaelt die Restzeit des Verzoegerungs-
        :                   auftrags
        :
```

6.13 OB 160 bis 163: Zählschleifen

Mit Hilfe dieser Sonderfunktions-Organisationsbausteine realisieren Sie Programmschleifen mit besonders günstiger Laufzeit.

Funktion

Die vier Organisationsbausteine OB 160, OB 161, OB 162 und OB 163 ermöglichen eine vierfache Schachtelung von Schleifen. Sie können damit in Systemdatenwörtern BS 60 bis 63 vier verschiedene Schleifenzähler einsetzen.

Jedem der vier OBs ist eine bestimmtes Systemdatenwort zugeordnet:

- BS 60: OB 160
- BS 61: OB 161
- BS 62: OB 162
- BS 63: OB 163

Programmieren der Programmschleife

In eines der Systemdatenwörter transferieren Sie den Wert für die erwünschte Anzahl an Schleifendurchläufen. Rufen Sie nun den dazugehörigen Sonderfunktions-OB auf, so wird der Schleifenzähler im Systemdatenwort um eins erniedrigt. Die Schleife wird so lange durchlaufen, bis der Schleifenzähler den Wert Null enthält.

> **Hinweis**
> Enthält der Schleifenzähler bereits vor Aufruf des Sonderfunktions-OBs den Wert Null, so wird er bei Aufruf ebenfalls um eins erniedrigt: Es erfolgen 65 536 Schleifendurchläufe!

Parameter

Systemdatenwörter BS 60 bis 63

Schleifenzähler,
mögliche Werte: 0 bis 65 535 dezimal (0 bis FFFFH)

Ergebnis

Schleifenzähler im Systemdatenwort > 0: VKE wird gesetzt (VKE = 1)

Schleifenzähler im Systemdatenwort = 0: VKE wird gelöscht (VKE = 0)

Die restlichen Bit- und Wortanzeigen werden immer gelöscht.

Die Akkumulatoren werden nicht verändert und nicht ausgewertet. Somit stehen diese zu Beginn des nächsten Schleifendurchlaufs noch zur Verfügung und müssen nicht neu geladen werden.

Fehlerfälle

keine

Beispiel

```
Zählschleife programmieren:

Im Merkerwort x steht die erwünschte Anzahl an Schleifendurchläufen.

                     :
Schleife initiali-
sieren:              :L    KB 0
                     :L    MW x            Schleifenzaehler
                     :!=   F
                     :SPB  =M002
                     :T    BS 62           Schleifenzaehler ins
                     :                     Systemdatenwort
                     :                     transferieren
                     :
"Schleifen-
programm":      M001 :  .
                     :  .
                     :  .
                     :  .
                     :
Schleife
verwalten:           :SPA  OB 162          Zaehlschleife
                     :SPB  =M001           bei VKE = 1 wird
                     :                     Schleife erneut
                     :                     durchlaufen
weiter:         M002 :  .
                     :  .
                     :  .
                     :  .
```

Ein weiteres Beispiel finden Sie im Abschnitt 9.2 "TNW und TNB: Speicherblöcke transferieren".

6.14 OB 170: Bausteinstack (BSTACK) lesen

Im Bausteinstack sind, ausgehend vom OB 1 bzw. FB 0, alle Bausteine eingetragen, die nacheinander aufgerufen worden sind und deren Bearbeitung noch nicht abgeschlossen ist.

Funktion

Mit Hilfe des Sonderfunktions-Organisationsbausteins OB 170 können Sie die im BSTACK vorhandenen Einträge in einen Datenbaustein einlesen. Auf diese Weise ermitteln Sie die vorhandene Anzahl an BSTACK-Einträgen und damit die Reserve, die Ihnen für weitere Einträge noch zur Verfügung steht.
Zu jedem Eintrag erhalten Sie die jeweilige Rücksprungadresse (STEP-Adreßzähler = SAZ), die absolute Anfangsadresse des in diesem Baustein gültigen Datenbausteins (DBA) sowie dessen Länge (Anzahl der Datenwörter = DBL).

Hinweis
Vor Aufruf des OB 170 muß ein **ausreichend langer** Datenbaustein (DB oder DX) aufgeschlagen werden. Für jeden gewünschten BSTACK- Eintrag benötigen Sie vier Datenwörter.

Parameter

Akkus

a) AKKU-2-L

Nummer des Datenworts (DW n), ab dem die Einträge im aufgeschlagenen DB abgelegt werden sollen (Offset)

b) AKKU-1-L

gewünschte Anzahl an BSTACK-Elementen,
zulässige Werte: 1 bis 62

Beispiel: Enthält der AKKU-1-L den Wert ’1’, erhalten Sie den letzten BSTACK-Eintrag, bei ’2’ den letzten und vorletzten, usw.

Ergebnis

Nach **erfolgreichem** Aufruf des OB 170

- steht im AKKU-2-L weiterhin der Offset im Datenbaustein,
- steht im AKKU-1-L die **tatsächlich** dargestellte Anzahl an BSTACK- Elementen [1],
- wird das VKE gelöscht,
- können die Ergebnisanzeigen ANZ 0 und ANZ 1 ausgewertet werden,
- sind alle übrigen Bit- und Wortanzeigen gelöscht.

[1] mögliche Werte: 0 - 62, wobei dargestellte Anzahl ≤ gewünschte Anzahl
0 = "kein BSTACK-Eintrag vorhanden" oder "Fehler"
(Inhalt von AKKU-1-L multipliziert mit 4 ergibt die Anzahl der beschriebenen Datenwörter im aufgerufenen DB!)

Beeinflussung der Ergebnisanzeigen VKE, ANZ 1 und ANZ 0

VKE	ANZ 1	ANZ 0	Abfrage mit	Bedeutung
0	0	1	SPM	vorhandene Anzahl BSTACK-Elemente < gewünschte Anzahl
0	0	0	SPZ	vorhandene Anzahl BSTACK-Elemente = gewünschte Anzahl
0	1	0	SPP	vorhandene Anzahl BSTACK-Elemente > gewünschte Anzahl
1	1	1	SPB	Fehler

Ablage der BSTACK-Elemente im aufgeschlagenen Datenbaustein

So werden die Inhalte des BSTACKs bei Aufruf des OB 170 im aufgerufenen Datenbaustein abgelegt (siehe auch Bild 6-3) :

A = BSTACK-Element-Nummer (62 bis 1)

(Bereits bei Ausgabe des letzten BSTACK-Elements läßt sich so die Reserve ermitteln: A = 17 Reserve = A - 1 = 16)

B = Tiefe des BSTACK-Elementes (1 bis 62)

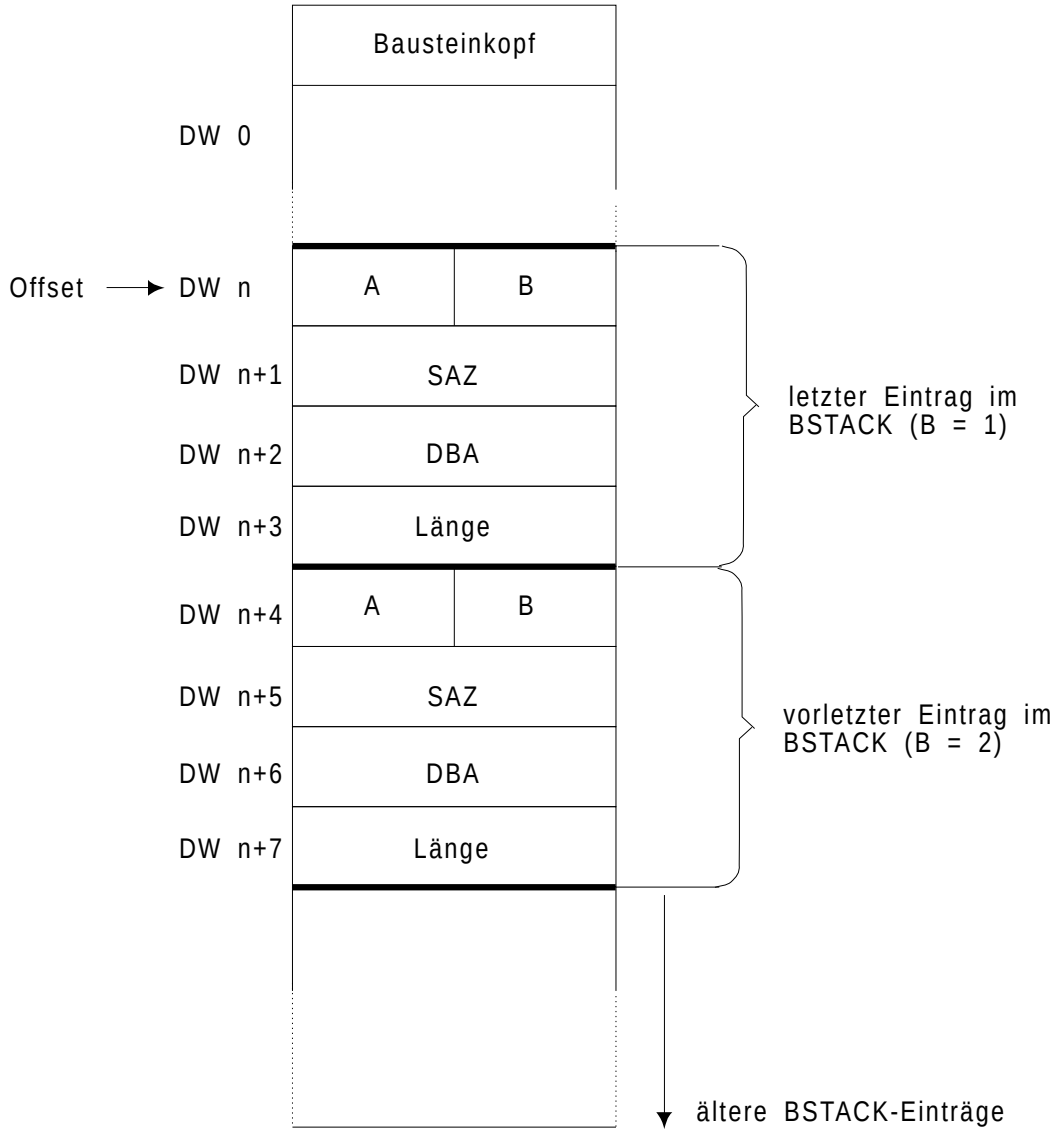


Bild 6-3 Ablage der BSTACK-Einträge in einem Datenbaustein

Fehlerfälle

- kein Datenbaustein aufgeschlagen
- aufgeschlagener Datenbaustein nicht vorhanden oder nicht ausreichend lang, um die gewünschte Anzahl von BSTACK-Einträgen aufnehmen zu können
- unzulässige Parameter in AKKU 1 und AKKU 2

Im Fehlerfall werden das VKE sowie die Ergebnisanzeigen ANZ 0 und ANZ 1 gesetzt (VKE, ANZ 0 und ANZ 1 = 1), die übrigen Bit- und Wortanzeigen werden gelöscht. Der Inhalt von AKKU-1-L wird zu ’0’.

Beispiel

```
Sie wollen die letzten 3 BSTACK-Einträge in den Datenbaustein DX 10 einlesen.
Die Einträge sollen im DX 10 ab Datenwort DW 16 abgelegt werden (siehe Bilder
6-4 und 6-5).

     :AX   DX    10         Aufschlagen DX 10
     :L    KY    0,16       BSTACK-Eintraege sollen ab DW 16 abgelegt werden
     :L    KY    0,3        gewünscht werden die letzten 3 BSTACK-Eintraege
     :SPA  OB    170

Im BSTACK sind 6 Bausteine eingetragen:
```

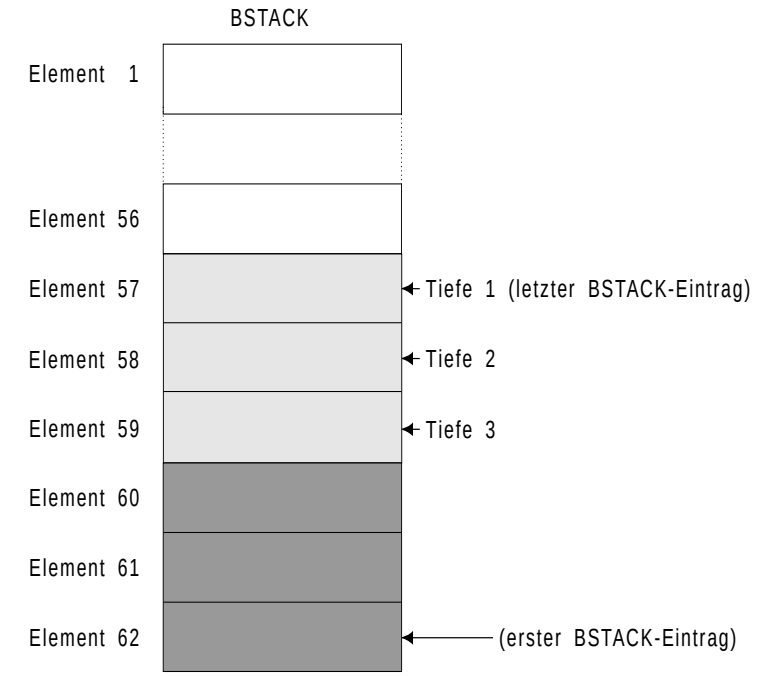


Bild 6-4 BSTACK-Belegung im Beispiel

Fortsetzung auf der nächsten Seite

Fortsetzung des Beispiels:

Nach Aufruf des OB 170 ist der Datenbaustein DX 10 folgendermaßen belegt:

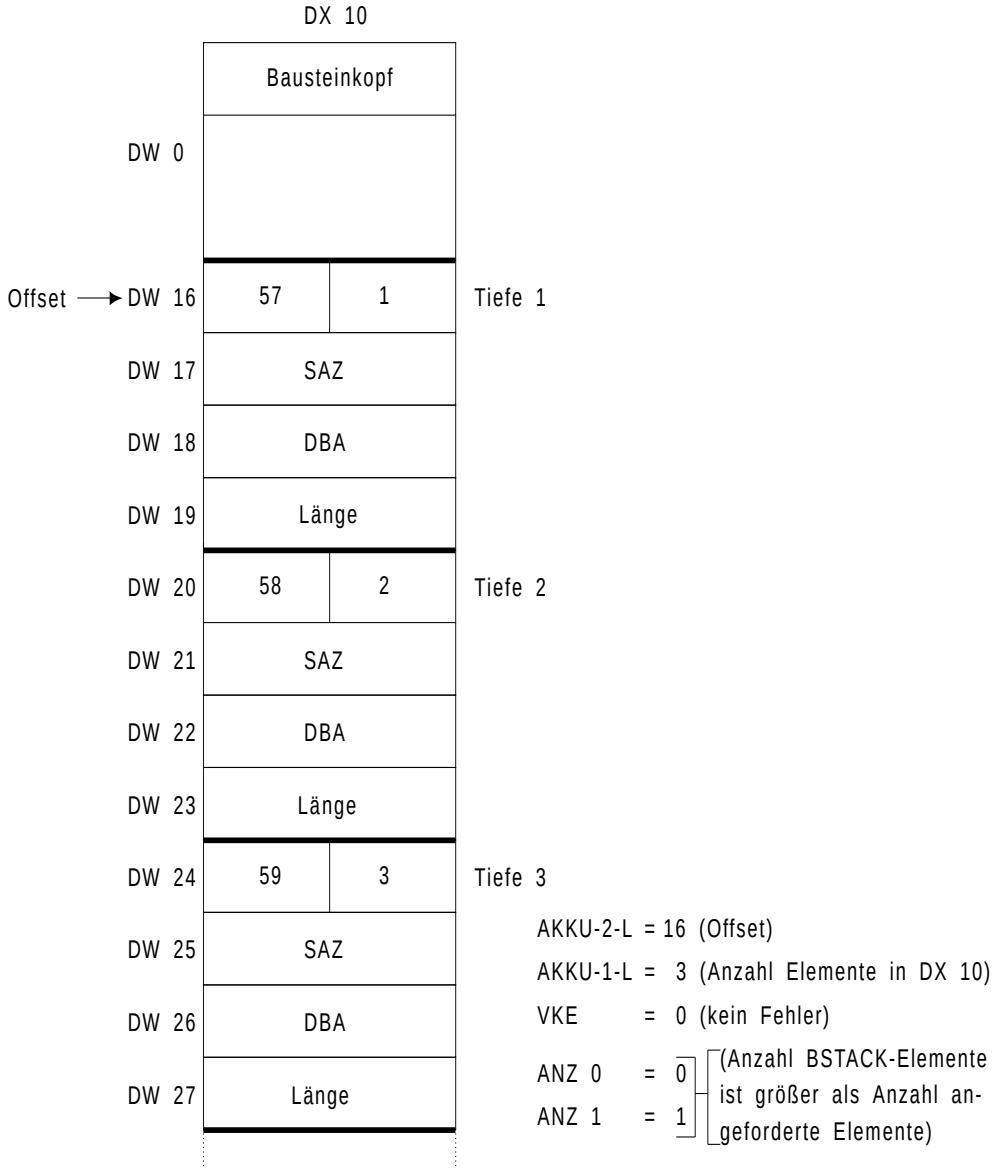


Bild 6-5 Belegung DX 10 im Beispiel nach Aufruf des OB 170

6.15 OB 180: Variabler Datenbaustein-Zugriff

DBA- DBL-Register

Beim Aufschlagen eines Datenbausteins mit den Operationen A DB und AX DX wird das 'DBA'-Register (**D**aten**b**austein-**A**nfangsadresse) mit der Adresse des Datenwortes DW 0 geladen, die im DB 0 hinterlegt ist.

Zugriffe auf Datenbausteine mit Opertationen wie L DR 60 oder B DW 240 usw. erfolgen immer relativ zur Datenbaustein-Anfangsadresse.

Zusätzlich zum DBA-Register wird bei jedem Aufruf eines Datenbausteins das 'DBL'-Register (**D**aten**b**austein-**L**änge) geladen: Es enthält die Länge (in Wörtern) des aufgeschlagenen DB- oder DX-Datenbausteins **ohne** Baustein-Kopf.

> **Hinweis**
> Im DBL-Register kann eine maximale Länge von bis zu 4091 Datenwörtern eingetragen sein!
> STEP-5-Zugriffe auf Datenwörter können nur auf Datenwort-Nummern bis 255 erfolgen.

Beispiel

Das DBA-Register enthält die Adresse des Speicherwortes, in welchem das DW 0 des DB 17 hinterlegt ist: DBA = **151BH**.

Im DBL-Register ist die Anzahl der Datenwörter hinterlegt: DBL = **8** (DW 0 bis DW 7).

Da der Zugriff auf Datenwörter mittels der STEP-5-Operationen L DW, U D, B DW usw. immer relativ zum DBA erfolgt, wird – um z.B. auf das DW 3 zuzugreifen – 3 zu 151BH addiert. Unter der Adresse 151EH ist das Datenwort DW 3 abgelegt.
Anhand des DBL-Registers wird geprüft, ob ein Transfer- oder Ladefehler vorliegt. So ist z.B. T DW 7 erlaubt, T DW 8 bzw. L DW 8 jedoch fehlerhaft.

Anwendung des OB 180

Der Sonderfunktions-OB 180 bietet Ihnen die Möglichkeit, auf strukturierte Daten in einem aufgeschlagenen Datenbaustein zuzugreifen. Dies können Sie dadurch erreichen, daß Sie die im Register DBA eingetragene Anfangsadresse des Datenbausteins mit Hilfe des OB 180 zum Ende des Datenbausteins hin verschieben. Gleichzeitig mit einer Verschiebung der Anfangsadresse wird durch OB 180 die im Register DBL eingetragene Bausteinlänge entsprechend vermindert. Dies ist wichtig, damit die CPU bei später erfolgenden Zugriffen auf den Datenbaustein eine Überwachung bei Lade- und Transferoperationen durchführen kann.

- Arbeiten mit Datenbausteinen, die eine Länge größer 261 Wörter (5 Wörter Kopf) haben: Mit Hilfe des OB 180 können Sie ein "Zugriffsfenster" von 256 Datenwörtern beliebig über einem Datenbaustein verschieben.

- Anwendung bei Datenstrukturen:
  Ein Datenbaustein kann in mehrere Datensätze mit gleicher Länge und gleicher Anordnung der darin enthaltenen Daten unterteilt sein. Man spricht in diesem Fall von einer Strukturierung des Datenbausteins.
  Ein so strukturierter Datenbaustein kann z. B. die Daten zu mehreren Teilprozessen enthalten, wobei im ersten Datenwort ein Temperaturwert, im zweiten ein Druck und in den übrigen Datenwörter andere Meßwerte des Teilprozesses abgelegt sind.
  Mit Hilfe des OB 180 können Sie auf die Daten jedes dieser Teilprozesse mit denselben Operationen (z. B. L DD, S D, T DR usw.) zugreifen, indem Sie das DBA-Register jeweils mit der Anfangsadresse für die Teilprozessdaten laden.

  Im Gegensatz zu anderen Substitutionsmechanismen (Substitution = indizierte Parametrierung) ergeben sich bei diesem Verfahren einfachere und laufzeitgünstigere STEP-5-Programme.

Funktion

Mit dem OB 180 wird die Anfangsadresse des aktuellen Datenbausteins um einen vorgegebenen Wert verschoben. Dabei wird berücksicht, daß die noch verfügbare Länge des DBs abnimmt (DBA- und DBL-Register werden entsprechend der Verschiebung geladen).

> **Hinweis**
> Vor Aufruf des OB 180 muß ein **ausreichend langer** Datenbaustein (DB oder DX) aufgeschlagen sein.

Parameter

AKKU-1-L

Versatz (Anzahl der Datenwörter, um die die Datenbausteinanfangs adresse verschoben werden soll),
zulässige Werte: $0 \leq$ AKKU-1-L $<$ DBL

Ergebnis

Nach **erfolgreichem** Aufruf des OB 180

- ist der Wert des DBA-Registers (= Adresse des DW 0) um den Wert des AKKU-1-L erhöht,
- ist der Wert des DBL-Registers um den Wert des AKKU-1-L erniedrigt,
- ist das VKE gelöscht (VKE = 0),
- sind alle übrigen Bit- und Wortanzeigen gelöscht.

Fehlerfälle

- negative Länge,
- kein Datenbaustein aufgeschlagen,
- Inhalt AKKU-1-L ≥ DBL.

Im Fehlerfall (Inhalt AKKU-1-L ≥DBL) bleiben DBA- und DBL- Register unbeeinflußt. Das VKE wird gesetzt (VKE = 1). Die übrigen Bit- und Wortanzeigen werden gelöscht.

Enthält das DBL-Register den Wert ’0’, so erkennt der OB 180, daß kein Datenbaustein aufgeschlagen ist. Das VKE wird gesetzt (VKE = 1) und signalisiert so einen Fehler.

DBA und DBL auf Anfangswert zurückstellen

Ein erneutes Aufschlagen des Datenbausteins mit den Operationen A DB oder AX DX stellt den Grundzustand wieder her.

Beispiel

```
Die Datenbausteinanfangsadresse (DBA = 151B)H im DB 17 (DBL = 8) soll um zwei
Datenwörter verschoben werden.

        :A    DB    17      DB 17 aufgeschlagen
        :L    KB    2       Verschiebung / Versatz als Konstante
        :SPA  OB    80      OB 180 aufrufen: DBA und DBL werden angepasst


Nach Aufruf des OB 180 läßt sich z.B. das unter der Adresse 1520H gespeicherte
Datenwort nicht mehr mit DW 5, sondern mit DW 3 ansprechen usw.. (siehe
Bild 6-6).
```

Fortsetzung des Beispiels:

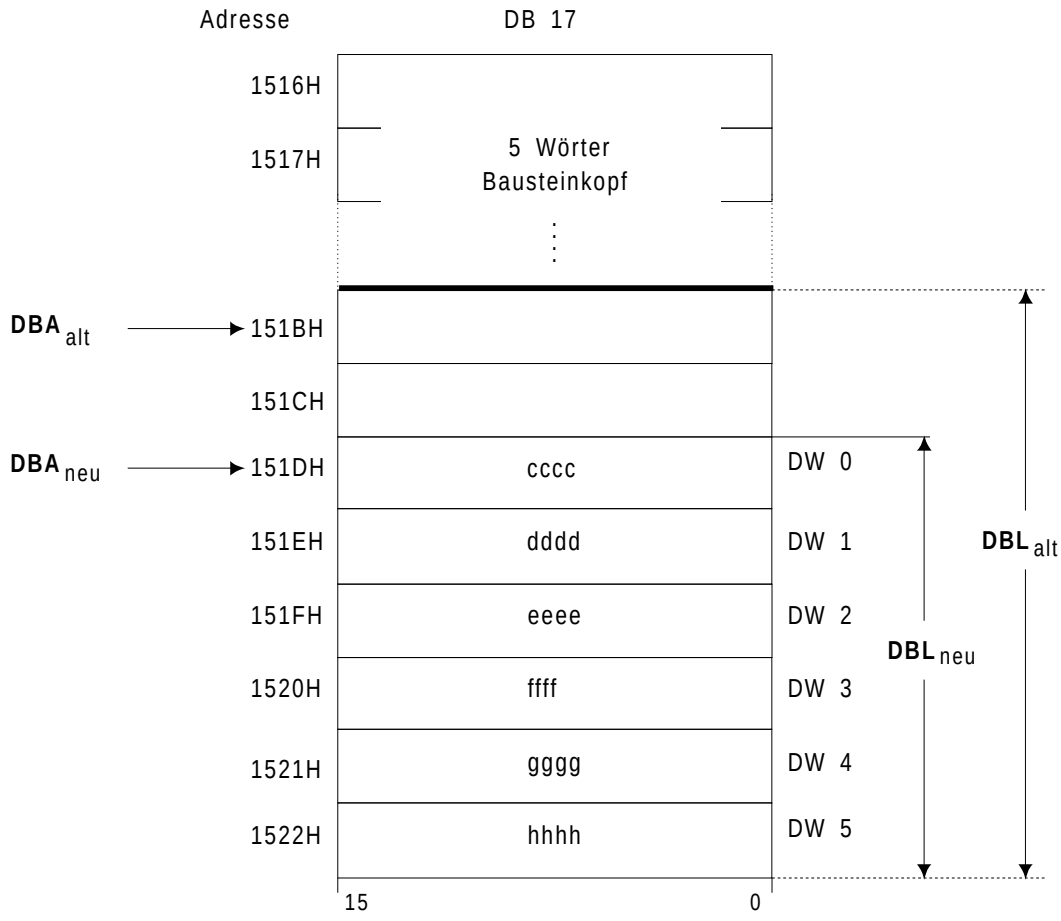


Bild 6-6 Verschieben der DB-Anfangsadresse

Aufgrund des gleichzeitig veränderten DBL-Registers bleibt die **Fehler-Überwachung gewährleistet:** Die Operation T DW 5 ist erlaubt, während T DW 6/L DW 6 fehlerhaft sind.

Durch erneute Aufrufe des OB 180 kann das DBA weiter erhöht werden (wobei das DBL weiter verringert wird): Die Operation A DB 17 stellt den Grundzustand (DBA = 151BH, DBL = 8) wieder her.
Hätte der DB 17 eine Länge von z.B. 258 Datenwörtern, könnten Sie mit STEP-5-Operationen nicht mehr auf DW 256 und DW 257 zugreifen. Durch Verschieben des DBA-Registers um 2 lassen sich die Datenwörter 256 und 257 mit "DW 254" und "DW 255" ansprechen.

(Zum DBA-/DBL-Register siehe auch Kapitel 9)

6.16 OB 181: Datenbausteine (DB/DX) testen

Mit dem Sonderfunktions-Organisationsbaustein OB 181 können Sie prüfen,

- ob ein bestimmter DB- oder DX-Datenbaustein vorhanden ist,
- unter welcher Adresse das erste Datenwort des Datenbausteins abgelegt ist,
- wieviele Datenwörter dieser Datenbaustein enthält,
- welcher Speichertyp und Speicherbereich (Anwenderspeicher: RAM oder EPROM, DB-RAM) benutzt wird.

Anwendung des OB 181

Eine Anwendung der Funktion "DB/DX testen" ist sinnvoll vor den Befehlen TNB/TNW, E DB/EX DX und vor Aufruf der Sonderfunktions- Organisationsbausteine OB 182, OB 254 und OB 255.

So können Sie beispielsweise vor einem Blocktransfer von Datenwörtern den OB 181 aufrufen, um sicherzustellen, daß der Zieldatenbaustein gültig und lang genug ist, um alle zu kopierenden Datenwörter aufzunehmen.

Funktion

Der OB 181 prüft, ob ein vorgegebener Datenbaustein vorhanden ist, und gibt als Ergebnis die charakteristischen Parameter des Datenbausteins zurück.

Parameter

AKKU-1-L

a) AKKU-1-LL

Bausteinnummer,
zulässige Werte: 1 bis 255

b) AKKU-1-LH

Bausteinkennung,
zulässige Werte: 1 = DB
2 = DX

Ergebnis

- Der geprüfte Baustein ist in der CPU **vorhanden:**

  - **AKKU-1-L**: Adresse des 1. Datenwortes (DW 0),
  - **AKKU-2-L**: Länge des Datenbausteins in Wörtern (ohne Bausteinkopf)
    Beispiel: In AKKU-2-L steht der Wert ’7’: Der Datenbaustein besteht aus den Datenwörtern DW 0 bis DW 6,
  - **VKE**: = 0,
  - **ANZ 1/ANZ 0**: werden entsprechend der Lage des Bausteins beeinflußt (siehe nachfolgende Tabelle),
  - **restliche Bit- und Wortanzeigen:** werden gelöscht.

- Der geprüfte Baustein ist in der CPU nicht **vorhanden** oder es ist ein Fehler aufgetreten:

  - **AKKU 1 und 2**: werden nicht verändert,
  - **VKE**: = 1,
  - **ANZ 1/ANZ 0**: = 1,
  - **restliche Bit- und Wortanzeigen:** werden gelöscht.

VKE, ANZ 1, ANZ 0

Entsprechend dem Prüfergebnis werden folgende Anzeigen gesetzt, die durch die in der Tabellenspalte "Abfrage" aufgeführten Operationen ausgewertet werden können:

VKE	ANZ 1	ANZ 0	Abfrage	Bedeutung		
0	0	1	SPM	DB/DX im Anwender-Modul	DB/DX im EPROM (nur lesbar)	DB/DX vorhanden
0	0	0	SPZ		DB/DX im RAM (schreib- und lesbar)	
0	1	0	SPP	DB/DX im DB-RAM		
1	1	1	SPB	DB/DX nicht vorhanden oder Fehler		

Fehlerfälle

- falsche Bausteinnummer (unzulässig: 0 – DB 0 oder DX 0),
- falsche Bausteinkennung (zulässig: 1 = DB, 2 = DX; unzulässig: 0, 3 bis 255),
- Speicherfehler.

Beispiele

siehe: Kapitel 8.3.2 / Kapitel 9.2 / Kapitel 9.3

6.17 OB 182: Datenbereich kopieren

Funktion

Der OB 182 kopiert einen Datenblock variabler Länge von einem Datenbaustein in einen anderen. Als Quell- und Zielbausteine können DB- und DX-Datenbausteine verwendet werden. Der Blockanfang im Quell- und Zieldatenbaustein ist frei wählbar. Der OB 182 kann maximal 4091 Datenwörter kopieren. Er enthält Pseudobefehlsgrenzen.

Hinweis
Quell- und Zielbaustein können identisch sein. Die Datenbereiche von Quelle und Ziel dürfen sich überlappen. Die **Originaldaten** des Quellbereichs werden auch bei Überlappung unverändert in den **Zielbereich** kopiert. Der **Überlappungsbereich in der Quelle** ist nach dem Kopiervorgang überschrieben. Diese Funktionseigenschaft können Sie einsetzen, um einen Datenbereich innerhalb eines Bausteins zu verschieben.

Parameter

1. Datenfeld mit Kopierparametern

Vor dem Aufruf des OB 182 versorgen Sie ein Datenfeld mit den erforderlichen Parametern für den gewünschten Kopiervorgang. Dieses Datenfeld kann in einem DB- oder DX-Datenbaustein, im M- oder S-Merkerbereich angelegt werden.

Das Datenfeld bezeichnet Quell- und Zieldatenbaustein, die Blockanfangsadresse in beiden Bausteinen sowie die Anzahl der zu übertragenden Datenwörter. Es besteht aus fünf Wörtern:

Bit-Nr.	15 8	7 0
1. Wort	Quell-DB-Typ	Quell-DB-Nr.
2. Wort	Nr. des 1. zu übertragenden Datenwortes im Quell-DB	
3. Wort	Ziel-DB-Typ	Ziel-DB-Nr.
4. Wort	Nr. des 1. zu übertragenden Datenwortes im Ziel-DB	
5. Wort	Anzahl der Datenwörter	

Die Parameter haben folgende Bedeutung und zulässige Wertebereiche:

Parameter	zulässiger Wertebereich
Datenbaustein-Typ (Quelle und Ziel)	1 = DB 2 = DX
Datenbaustein-Nr. (Quelle und Ziel)	3 bis 255
Nr. des 1. Datenwortes (Quelle und Ziel)	0 bis 4090
Anzahl der Datenwörter	1 bis 4091

Datenfeld im Merkerbereich

Wenn Sie das Datenfeld in einem Merkerbereich anlegen, müssen Sie folgende Zuordnung der Datenfeldwörter zu den Merkerbytes berücksichtigen. Dabei ist 'x' der Parameter "Nr.des 1. Datenfeldwortes", den Sie beim Aufruf des OB 182 im AKKU-1-L hinterlegen müssen:

Bit-Nr.	15 8	7 0
1. Datenfeldwort	Merkerbyte x	Merkerbyte x+1
2. Datenfeldwort	Merkerbyte x+2	Merkerbyte x+3
3. Datenfeldwort	Merkerbyte x+4	Merkerbyte x+5
4. Datenfeldwort	Merkerbyte x+6	Merkerbyte x+7
5. Datenfeldwort	Merkerbyte x+8	Merkerbyte x+9

2. Akkus

2a) AKKU-2-L

Der AKKU-2-L enthält Angaben zum verwendeten Datenfeld. Er muß folgenden Aufbau haben:

Bit-Nr.	15 8	7 0
	Adreßbereichs-Typ	Datenbaustein-Nr.

Parameter im AKKU-2-L

Adreßbereichs-Typ,
zulässige Werte:
1 = DB-Datenbaustein
2 = DX-Datenbaustein
3 = M-Merkerbereich
4 = S-Merkerbereich

Datenbaustein-Nr.,
zulässige Werte: 3 bis 255 (nur bei Adreßbereichs-Typ '1' oder '2'; bei Adreßbereichs-Typ '3' oder '4' irrelevant)

2b) AKKU-1-L

Nummer des 1. Datenfeldwortes, mögliche Werte (in Abhängigkeit vom Adreßbeeichs-Typ:

DB, DX:	0 bis 2043
M-Merker:	0 bis 246 (= Nr. Merkerbyte 'x ')
S-Merker	0 bis 1014 (= Nr. Merkerbyte 'x ')

Ergebnis

Nach korrekter Bearbeitung des OB 150 sind die Anzeigenbits OR, ERAB und OS = 0. Alle anderen Anzeigenbits sowie AKKU 1 und AKKU 2 sind unverändert.

Fehlerfälle

Im Fehlerfall werden der **OB 19** oder **OB 31** aufgerufen. Sind der OB 19 oder OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen werden in AKKU 1 und AKKU 2 Fehlerkennungen hinterlegt (siehe nachfolgende Tabelle).

Tabelle 6-9 Fehlerkennungen des OB 182

AKKU-1-L	AKU-2-L	Bedeutung	aufger. OB
1A06H	–	Datenbaustein nicht geladen	OB 19
1A34H	0001H	Beschreibung des Datenfeldes fehlerhaft	OB 31
	0100H	Adreßbereich-Typ unzulässig	
	0101H	Datenbaustein-Nr. unzulässig	
	0102H	"Nummer des ersten Datenfeldwortes" unzulässig	
	0200H	"Quell-Datenbaustein-Typ" unzulässig	
	0201H	"Quell-Datenbaustein-Nr." unzulässig	
	0202H	"Nr. des 1. zu übertragenden Datenwortes im Quell-DB" unzulässig	
	0203H	Länge Quelldatenbaustein im Bausteinkopf < 5 Wörter	
	0210H	"Ziel-Datenbaustein-Typ" unzulässig	
	0211H	"Ziel-Datenbaustein-Nr." unzulässig	
	0212H	"Nr. des 1. zu beschreibenden Datenwortes im Ziel-DB" unzulässig	
	0213H	Länge Zieldatenbaustein im Bausteinkopf < 5 Wörter	
	0220H	"Anzahl zu übertragende Datenwörter" unzulässig (= 0 oder > 4091)	
	0221H	Quelldatenbaustein zu kurz	
	0222H	Zieldatenbaustein zu kurz	
	0223H	Zieldatenbaustein liegt im EPROM	

6.18 OB 190/192: Merker in Datenbaustein übertragen

Anwendung

Die Organisationsbausteine OB 190 und OB 192 übertragen eine vom Anwender vorgegebene Anzahl Merkerbytes in einen dafür vorgesehenen Datenbaustein.
Dies kann z.B. von Vorteil sein vor Bausteinaufrufen, in Fehler-Organisationsbausteinen oder bei Unterbrechung der zyklischen Programmbearbeitung durch eine zeit- oder alarmgesteuerte Programmbearbeitung.
Mit Hilfe der Organisationsbausteine OB 191 und OB 193 können Sie diese Merkerbytes anschließend wieder aus dem Datenbaustein zurückschreiben.

Hinweis
Verwenden Sie OB 190 und OB 191 für das einfache Retten und Zurücklesen der Merkerbytes, da Sie damit erhebliche Laufzeitvorteile gewinnen.

Vor dem Aufruf von OB 190/192 muß ein Datenbaustein (DB/DX) aufgeschlagen werden!

Die OB 190/192 übertragen **nur aus dem M-Merkerbereich**, jedoch **nicht** aus dem **S-Merkerbereich** in einen Datenbaustein.

Funktion

Nach Aufruf des OB 190/192 werden im aufgeschlagenen Datenbaustein die Merkerbytes ab der angegebenen Datenwortadresse gespeichert. Den Bereich der zu rettenden Merker entnehmen OB 190/192 dem AKKU 2.
OB 190 und OB 192 sind identisch mit Ausnahme der Art und Weise, in der sie die Merkerbytes übertragen:

OB 190 überträgt die Merker **byteweise.**

OB 192 überträgt die Merker **wortweise.**

Dies ist von Belang, wenn die in den Datenbaustein übertragenen Daten anschließend bearbeitet werden sollen und der Datenbaustein nicht nur als einfacher Zwischenspeicher benutzt wird.

Die folgende Abbildung soll diesen Unterschied verdeutlichen:

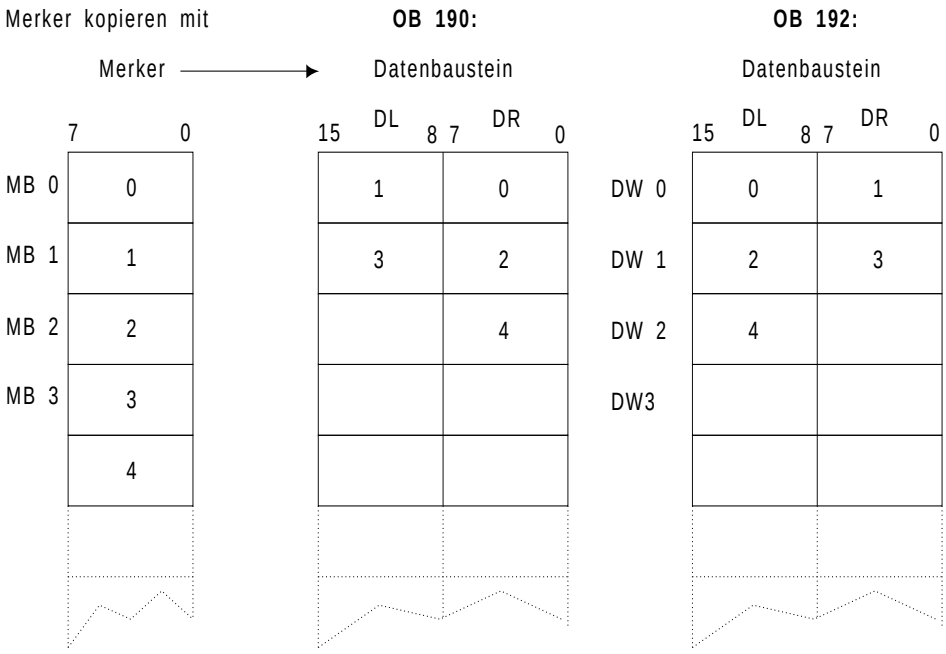


Bild 6-7 Byteweises (OB 190) und wortweises (OB 192) Übertragen

Hinweis
Falls eine **ungerade** Anzahl von Merkerbytes übertragen wird, so wird das **letzte** benutzte Datenwort des Datenbausteins **nur zur Hälfte** genutzt. Bei **OB 190** bleibt das Datum **links**, bei **OB 192** das Datum **rechts** im Ziel-DB **unverändert**.

Parameter

1. Angaben zur Quelle:

1a) AKKU-2-LH

Erstes zu übertragendes Merkerbyte,
zulässige Werte: 0 bis 255

1b) AKKU-2-LL

Letztes zu übertragendes Merkerbyte,
zulässige Werte: 0 bis 255

(Letztes Merkerbyte ≥ Erstes Merkerbyte !)

2. Angaben zum Ziel

AKKU-1-L

Nummer des ersten zu beschreibenden Datenwortes im aufgeschlagenen Datenbaustein:

Die zulässigen Werte orientieren sich an der Länge des Datenbausteins im Speicher. Es können dabei Nummern > 255 auftreten.

Ergebnis

Wird der Sonderfunktions-OB 190/192 **korrekt** bearbeitet, so wird das VKE gelöscht (VKE = 0). Die AKKUs bleiben unverändert.

Im **Fehlerfall** wird das VKE gesetzt (VKE = 1), die AKKUs bleiben unverändert.

Fehlerfälle

- kein DB- oder DX-Datenbaustein aufgeschlagen,
- falscher Merkerbereich (Letztes Merkerbyte < Erstes Merkerbyte),
- Datenwort-Nummer nicht vorhanden,
- Länge des DB- oder DX-Datenbausteins nicht ausreichend.

6.19 OB 191/193: Datenblöcke in Merkerbereich übertragen

Anwendung

Mit Hilfe der Organisationsbausteine OB 191 und OB 193 können Sie Daten aus einem Datenbaustein in den Merkerbereich übertragen. So können beispielsweise die zuvor in einen Datenbaustein "geretteten" Merkerbytes wieder in den Merkerbereich zurückgeschrieben werden.

OB 191/193 unterscheiden sich von den Organisationsbausteinen OB 190/192 nur dadurch, daß Quelle und Ziel vertauscht sind:

OB 190/192: Merkerbereich ——→ Datenbaustein

OB 191/193: Merkerbereich ←—— Datenbaustein

Hinweis
Vor dem Aufruf von OB 191/193 muß ein **ausreichend langer** Datenbaustein (DB/DX) aufgeschlagen werden!

Die OB 191/193 übertragen aus dem Datenbaustein **nur in den M-Merkerbereich**, jedoch **nicht** in den **S-Merkerbereich** .

Funktion

Nach Aufruf des OB 191/193 werden aus dem aufgeschlagenen Datenbaustein Datenwörter ab der angegebenen Datenwortadresse gelesen und in den Merkerbereich übertragen.

OB 191 und OB 193 sind identisch mit Ausnahme der Art und Weise, in der sie die Daten übertragen:

OB 191 überträgt die Datenwörter **byteweise.**

OB 193 überträgt die Datenwörter **wortweise**.

Die Abbildung auf der folgenden Seite soll Ihnen diesen Unterschied verdeutlichen.

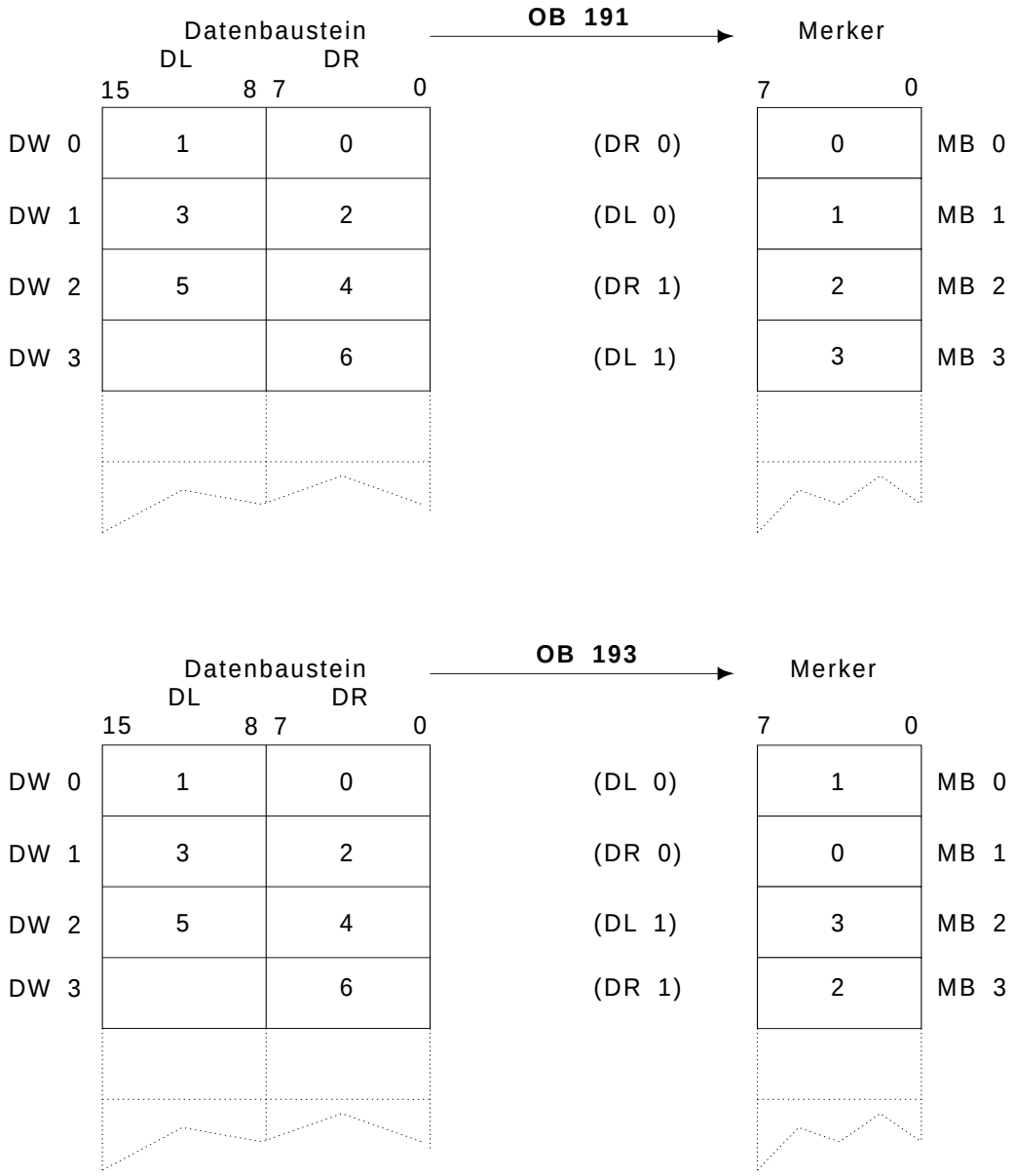


Bild 6-8 Byteweises (OB 191) und wortweises (OB 193) Übertragen

Parameter

1. Angaben zur Quelle:

1a) AKKU-2-L

Nummer des ersten zu übertragenden Datenwortes im aufgeschlagenen Datenbaustein

2. Angaben zum Ziel

2a) AKKU-1-LH

Erstes zu beschreibendes Merkerbyte,
zulässige Werte: 0 bis 255

2b) AKKU-1-LL

Letztes zu beschreibendes Merkerbyte,
zulässige Werte: 0 bis 255

(Letztes Merkerbyte ≥ Erstes Merkerbyte !)

Ergebnis

Wird der Sonderfunktions-OB 191/193 **korrekt** bearbeitet, so wird das VKE gelöscht (VKE = 0). Die AKKUs bleiben unverändert.

Im **Fehlerfall** wird das VKE gesetzt (VKE = 1), die AKKUs bleiben unverändert.

Fehlerfälle

- kein DB- oder DX-Datenbaustein aufgeschlagen,
- falscher Merkerbereich (Letztes Merkerbyte < Erstes Merkerbyte),
- Datenwort-Nummer nicht vorhanden,
- Länge des DB- oder DX-Datenbausteins nicht ausreichend.

Beispiele

Beispiel 1:

Vor Aufruf des Programmbausteins PB 12 sind alle Merker (MB 0 bis MB 255) in den Datenbaustein DX 37 ab Adresse 100 zu retten und anschließend wieder zurückzuschreiben.

```
Retten:       :AX   DX    37                Datenbaustein aufrufen
              :L    KY    0,255             Merkerbereich MB 0 bis MB 255
              :L    KB    100               Nummer des 1. Datenworts im Ziel
              :SPA  OB    190               Merker retten

Baustein-
wechsel:      :SPA  PB    12

Zurück-
schreiben:    :                             (Datenbaustein bereits aufgerufen)
              :L    KB    100               Nummer des 1. Datenworts in der
              :                             Quelle
              :L    KY    0,255             Merkerbereich MB 0 bis MB 255
              :SPA  OB    191               Merker zurückschreiben
```

Beispiel 2:

Merker, die vom zyklischen Anwenderprogramm benutzt werden, können nicht zusätzlich durch ein zeit- oder alarmgesteuertes Anwenderprogramm genutzt werden. Jeder Programmbearbeitungsebene muß ein bestimmter Teil des Merkerbereichs zugeordnet sein.

```
Z.B.: zyklisches Anwenderprogramm:          MB 0   ... . ... MB 99

      zeitgesteuertes Anwenderprogramm:     MB 100 ... . ... MB 199

      alarmgesteuertes Anwenderprogramm:    MB 200 ... . ... MB 255
```

Falls jedoch das zyklische Anwenderprogramm bereits alle 256 Merkerbytes benutzt und beispielsweise das zeitgesteuerte Anwenderprogramm ebenfalls alle 256 Merkerbytes benötigt, müssen die Merker beim Wechsel der Bearbeitungsebene ausgetauscht und zwischengespeichert werden.

Am schnellsten können die Merker mit Hilfe der Sonderfunktionen OB 190 und OB 191 gerettet und geladen werden. Bild 6-9 zeigt, wie ein von OB 1 und OB 13 (100 ms Weckalarm) gemeinsamer Merkerbereich MBx bis MBy in einem Datenbaustein DBz zwischengespeichert werden:

Fortsetzung auf der nächsten Seite

Fortsetzung von Beispiel 2:

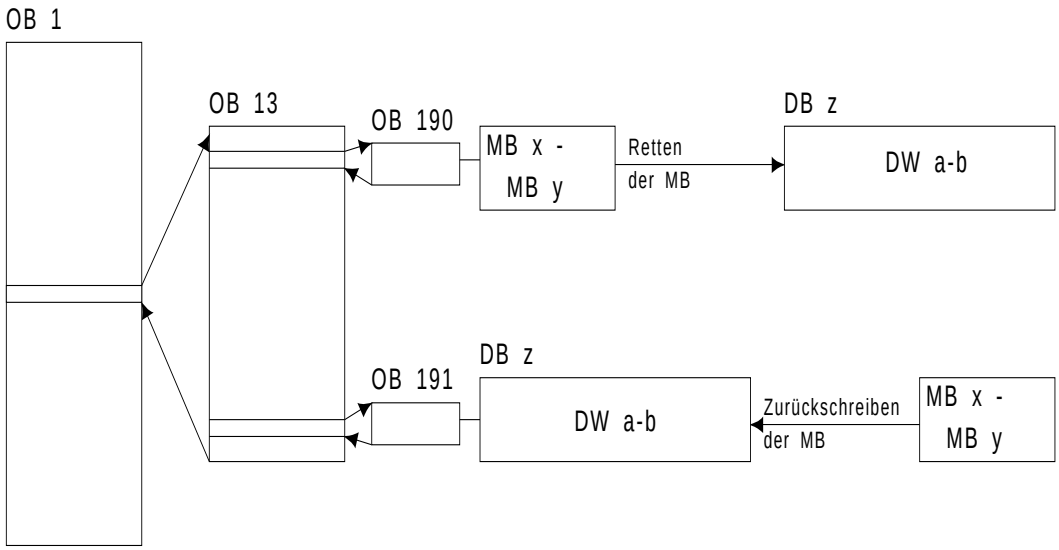


Bild 6-9 Retten von Merkerbereichen bei Wechsel der Programmbearbeitungsebene

STEP-5-Programm im OB 13:

```
:A    DB    100
:L    KY    0,255
:L    KB    0
:SPA  OB    190
:L    KB    128
:L    KY    0,255
:SPA  OB    191
:
:
:A    DB    100
:L    KY    0,255
:L    KB    128
:SPA  OB    190
:L    KB    0
:L    KY    0,255
:SPA  OB    191
:BE
```

Weitere Anwendungen für die Organisationsbausteine OB 190 bis 193

- Bei der CPU 928B werden Operationen zur Einzelbitverarbeitung (U, O, ON, UN, S, R, =), die auf den Merkerbereich zugreifen, wesentlich schneller bearbeitet als vergleichbare Operationen, die auf Datenbausteine zugreifen (vergleichen Sie hierzu z. B. die Operationen 'U M' ⟷ 'U D' oder 'S M' ⟷ 'S D'!).

  Aus diesem Grund verbessern Sie die Laufzeit, wenn Sie die Daten in den Merkerbereich kopieren, diese dort bearbeiten und anschließend wieder in den Datenbaustein zurückübertragen.

- Ohne großen Aufwand lassen sich im Datenbaustein High-Byte und Low-Byte vertauschen, indem mit den entsprechenden OBs die Datenwörter in den Merkerbereich und wieder zurück übertragen werden, wie in Bild 6-10 dargestellt:

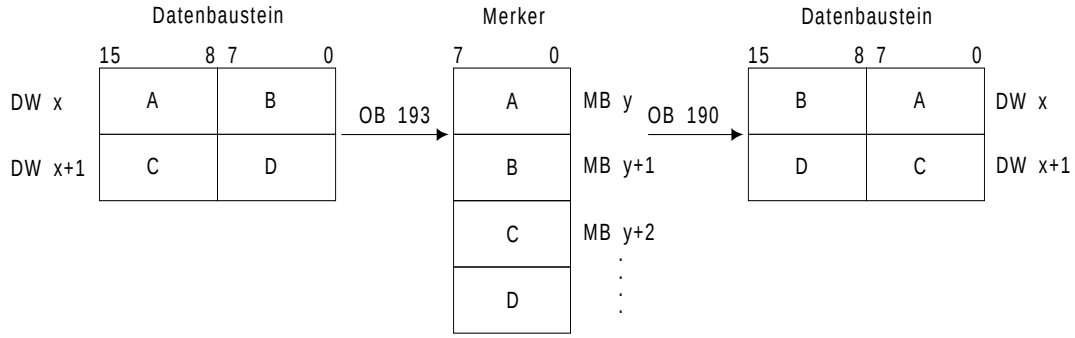


Bild 6-10 Vertauschen von High-Byte und LOW-Byte in einem DB mit Hilfe von OB 193/OB190

- Sie können Datenblöcke innerhalb eines Datenbausteins "verschieben", wenn Sie beim Zurückübertragen aus dem Merkerbereich zwar dieselbe DB-Nummer, jedoch ein anderes Datenwort angeben.

6.20 OB 202 bis 205: Mehrprozessor-Kommunikation

Eine detaillierte Anleitung zu diesen Sonderfunktions-Organisationsbausteinen finden Sie in Kapitel 10.

Die Sonderfunktions-Organisationsbausteine OB 200 und OB 202 bis OB 205 ermöglichen im Mehrprozessorbetrieb Datenübertragungen zwischen den einzelnen CPUs unter Verwendung des Koordinators KOR 923C.

- **OB 200: Initialisieren**

  Dieser Sonderfunktions-Organisationsbaustein richtet im Koordinator KOR 923C den Speicher ein, in dem die zu übertragenden Datenblöcke zwischengespeichert werden.

- **OB 202: Senden**

  Diese Funktion übergibt einen Datenblock in den Zwischenspeicher des KOR 923C und gibt an, wieviele Datenblöcke noch gesendet werden können.

- **OB 203: Sende-Test**

  Der Sonderfunktions-OB 203 ermittelt die Anzahl der freien Speicherblöcke im Zwischenspeicher des Koordinators KOR 923C.

- **OB 204: Empfangen**

  Diese Funktion übernimmt einen Datenblock vom Zwischenspeicher des KOR 923C und zeigt an, wieviele Datenblöcke noch empfangen werden können.

- **OB 205: Empfangs-Test**

  Der Sonderfunktions-Organisationsbaustein OB 205 ermittelt die Anzahl belegter Speicherblöcke im Zwischenspeicher des KOR 923C.

6.21 OB 216 bis 218: Kachelzugriffe

Was sind Kacheln?

Um im Adreßbereich des S5-Busses eine große Anzahl von Kommunikationsspeichern unterbringen zu können, ist ein Adreßbereich mit einer Länge von 1024 bytes (reserviert sind 2048 bytes) 256-mal auf den Speicher abgebildet. Weil diese 256 Abbildungen wie einzelne "Kacheln" neben- oder hintereinander liegen, werden diese Speicherbereiche auch "Kachelspeicher" genannt.

Im Mehrprozessorbetrieb darf von allen beteiligten Baugruppen zu jeder Zeit immer nur auf **eine** Kachel dieses Kachelspeichers zugegriffen werden, alle übrigen Kacheln müssen für das Lesen oder Schreiben gesperrt sein. Dies wird so realisiert:

Zur Anwahl der benötigten Kachel dient ein "Kacheladreßgister", das auf allen mit Kacheln arbeitenden Baugruppen vorhanden ist und eine feste Adresse auf dem S5-Bus hat. Auf jeder dieser Baugruppen werden durch den Anwender per DIL-Schalter die Nummern (Adressen) der Kacheln eingestellt, so daß jede Kachel nur einmal im AG vorhanden ist.

Die CPU gibt vor jedem Lesen oder Schreiben einer Kachel durch Schreiben in das Kacheladreßregister die Kachelnummer an. Alle nach diesem Verfahren arbeitenden Baugruppen auf dem S5-Bus empfangen **gleichzeitig** ("broadcast") diese Nummer und legen sie in ihrem Speicher ab. Nur die so adressierte Kachel ist nun über den S5-Bus schreib- und lesbar, alle übrigen Kacheln sind gesperrt.

Wie können Sie auf Kacheln zugreifen?

Die Organisationsbausteine OB 216 bis OB 218 sowie einige STEP-5-Operationen (siehe Kapitel 9) ermöglichen den Zugriff auf sogenannte "Kacheln".

Die Organisationsbausteine enthalten folgende Funktionen:

- **OB 216:**

  Schreiben eines Bytes/Wortes/Doppelwortes auf eine Kachel

- **OB 217:**

  Lesen eines Bytes/Wortes/Doppelwortes von einer Kachel

- **OB 218:**

  Belegen einer Kachel durch die CPU (dient der Koordinierung im Mehrprozessorbetrieb)

Diese Funktionen dienen Testzwecken und ermöglichen die Programmierung von Hantierungsbausteinen oder ähnlichen Funktionen.

Hinweis
In der Regel können Sie alle Funktionen mit Hilfe der Standard-Funktionsbausteine "Hantierungsbausteine" und der integrierten Sonderfunktions-Organisationsbausteine "Mehrprozessor-Kommunikation" (OB 200, OB 202 bis OB 205) ausführen, mit denen alle Kachelzugriffe "automatisch" abgewickelt werden.

Kachelzugriffe sollten Sie möglichst nur durch Aufrufe der OB 216 bis OB 218 programmieren. Die dafür auch zur Verfügung stehenden STEP-5-Operationen sollten Sie dagegen nur verwenden, wenn Sie über sehr gute Systemkenntnisse verfügen.

Adreßbereiche für Peripherie auf dem S5-Bus

Kachelgröße	Belegter Adressraum
1024 Adressen (Byte- oder Wortadressen)	F400H - F7FFH
2048 Adressen (Byte- oder Wortadressen)	F400H - FBFFH

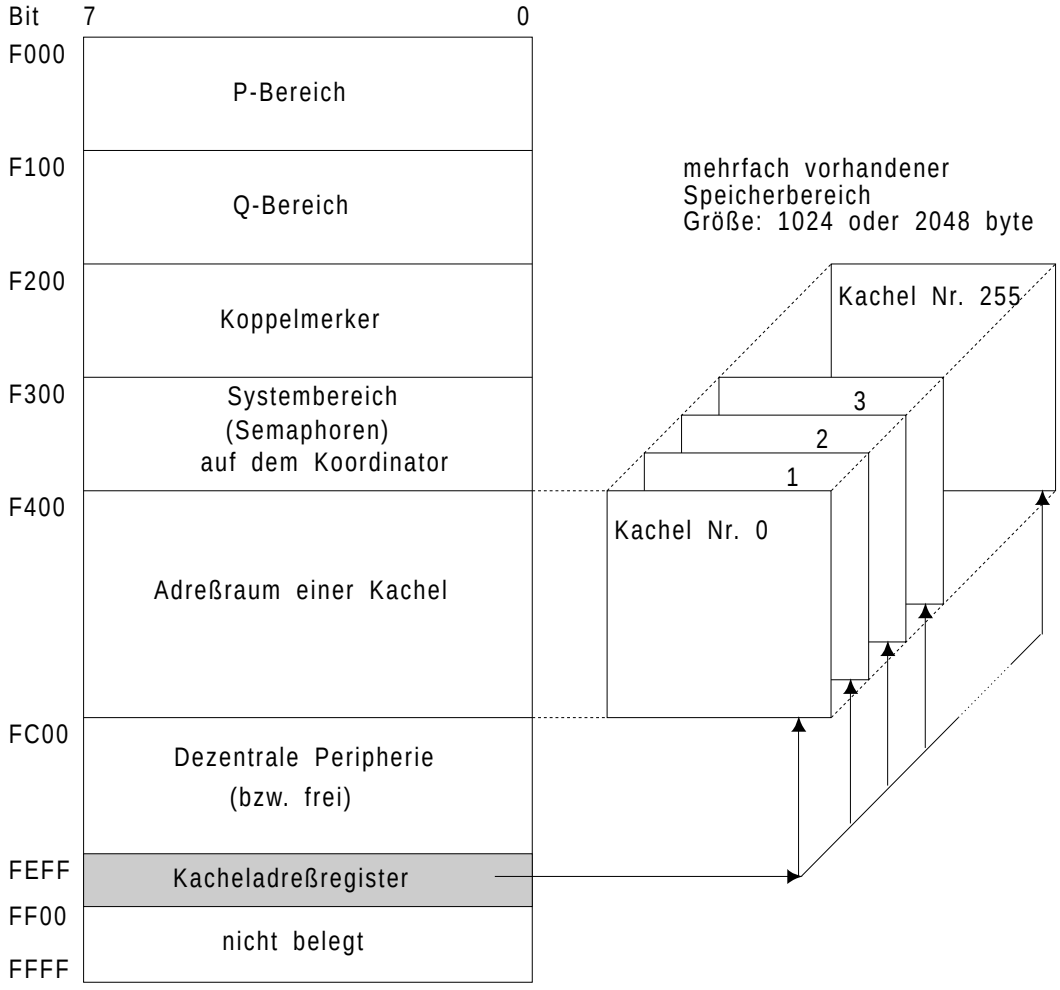


Bild 6-11 Lage des Kacheladreßraums auf dem S5-Bus

Welche der insgesamt 256 Kacheln verwendet werden soll, geben Sie bei der Parametrierung der Sonderfunktions-Organisationsbausteine OB 216, OB 217 und OB 218 an. Die Nummer der "aktuellen" Kachel wird daraufhin automatisch in eine Zelle mit der Adresse 0FEFFH eingetragen (siehe Bild 6-11). Alle Datenübertragungen beziehen sich dann auf die Kachel, deren Nummer eingetragen wurde.

Hinweis
Das Kacheladreßregister mit der Adresse 0FEFF H ist nicht lesbar. Sie können jedoch unter dieser Adresse auf der Koordinatorbaugruppe 923C das Busfehlerregister auslesen (siehe Systemhandbuch AG 135U/155U)).

Hinweise zur Parametrierung

Beim Schreiben (OB 216) und Lesen (OB 217) eines Bytes/Wortes/Doppelwortes auf einer Kachel werden die Bytes in folgender Reihenfolge angesprochen:

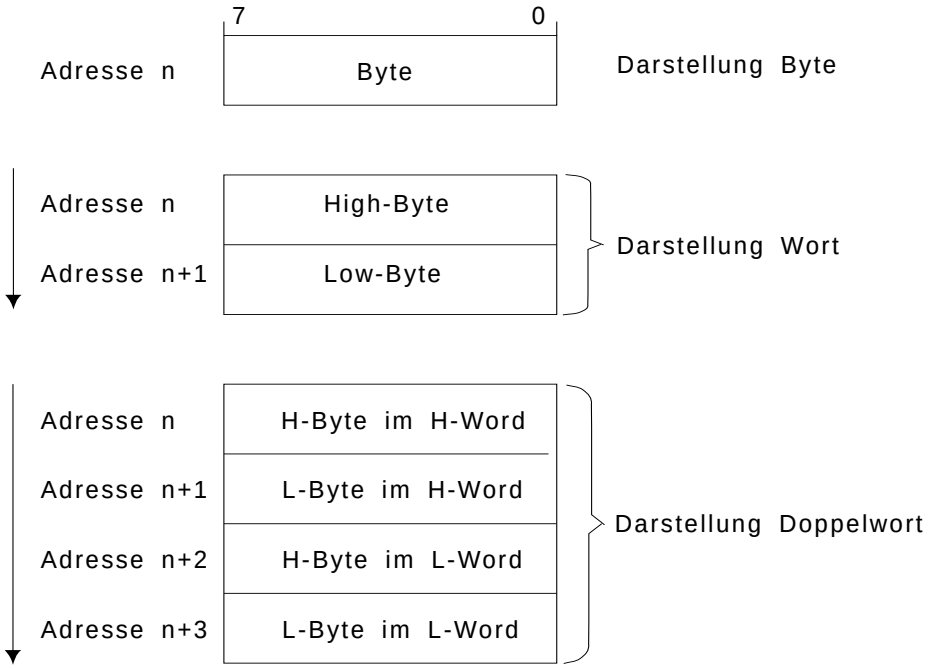


Bild 6-12 Lage der Bytes beim Schreiben (OB 216) / Lesen (OB 217) als Wort oder Doppelwort auf einer Kachel

6.21.1 OB 216: Schreiben auf eine Kachel

Funktion

Der Sonderfunktions-Organisationsbaustein überträgt ein Byte, Wort oder Doppelwort vom AKKU 1 (rechtsbündig) zu einer bestimmten Kachel.
Das **Adressieren der Kachel** im Ein- oder Mehrprozessorbetrieb und das **Übertragen des vollständigen Datums** (1, 2 oder 4 byte) bilden eine **untrennbare Programmeinheit**, die nicht unterbrochen werden kann.

Parameter

Akkus

a) AKKU-3-LH

Kennung des zu übertragenden Datums,
zulässige Werte:
0 = Byte
1 = Wort
2 = Doppelwort

b) AKKU-3-LL

aktuelle Kachel-Nummer ,
zulässige Werte: 0 bis 255

c) AKKU-2-L

Zieladresse auf der Kachel,
zulässige Werte: 0 bis 2047

d) AKKU 1

Datum, das geschrieben werden soll
(Byte, Wort, Doppelwort: rechtsbündig)

Akku-Belegung **vor** dem Schreiben:

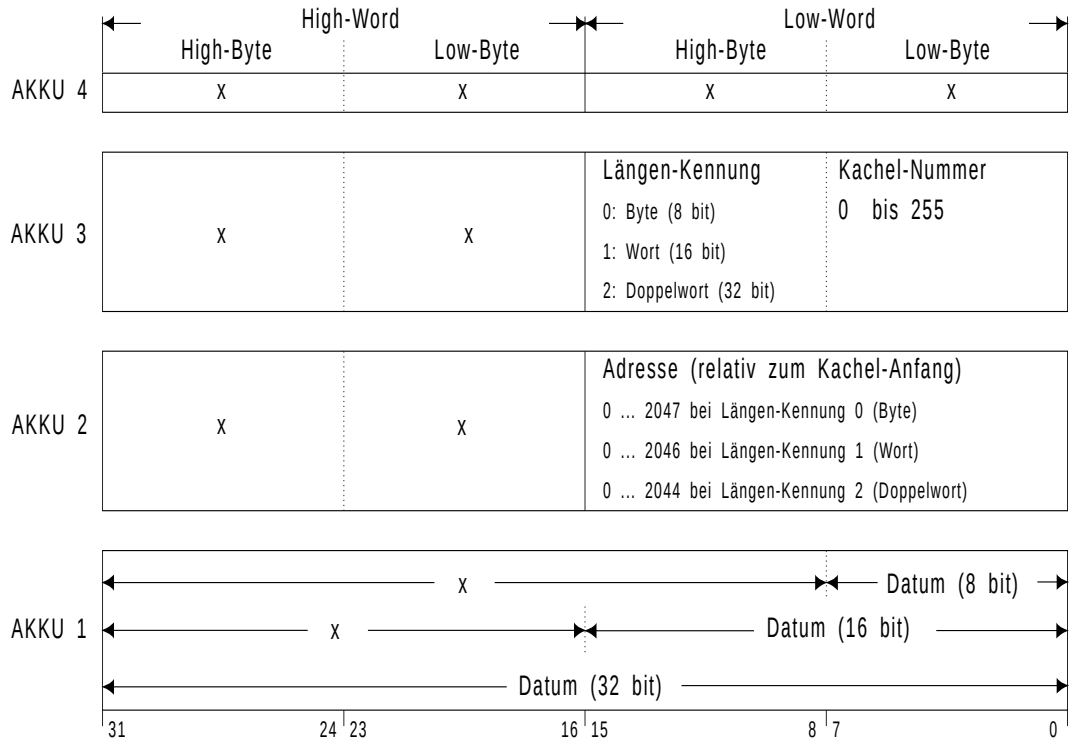


Bild 6-13 Inhalte der AKKUs vor Aufruf des OB 216

Ergebnis

- Das Schreiben auf die Kachel verläuft **erfolgreich:**
  - **AKKU 1 und AKKU 3:** werden nicht verändert,
  - **AKKU-2-L:** enthält einen um 1, 2 oder 4 erhöhten Wert (je nach Länge des übertragenen Datums),
  - **VKE:** = 1,
  - **restliche Bit und Wort-anzeigen:** werden gelöscht.
- Das Schreiben auf die Kachel ist **nicht möglich**:
  - **alle Akkus:** werden nicht verändert,
  - **VKE:** = 0,
  - **restliche Bit- und Wort-anzeigen:** werden gelöscht.

Fehlerfälle

- falsche Längenkennung in AKKU-3-LH,
- Zieladresse auf der Kachel falsch oder nicht vorhanden,
- angegebene Kchel-Nr. nicht vorhanden.

6.21.2 OB 217: Lesen aus einer Kachel

Funktion

Der Sonderfunktions-Organisationsbaustein überträgt ein Byte, Wort oder Doppelwort von einer bestimmten Kachel zum AKKU 1 (rechtsbündig).
Das **Adressieren der Kachel** im Ein- oder Mehrprozessorbetrieb und das **Übertragen des vollständigen Datums** (1, 2 oder 4 byte) bilden eine **untrennbare Programmeinheit**, die nicht unterbrochen werden kann.

Parameter

Akkus

a) AKKU-3-LH

Kennung des zu übertragenden Datums,
zulässige Werte:
0 = Byte
1 = Wort
2 = Doppelwort

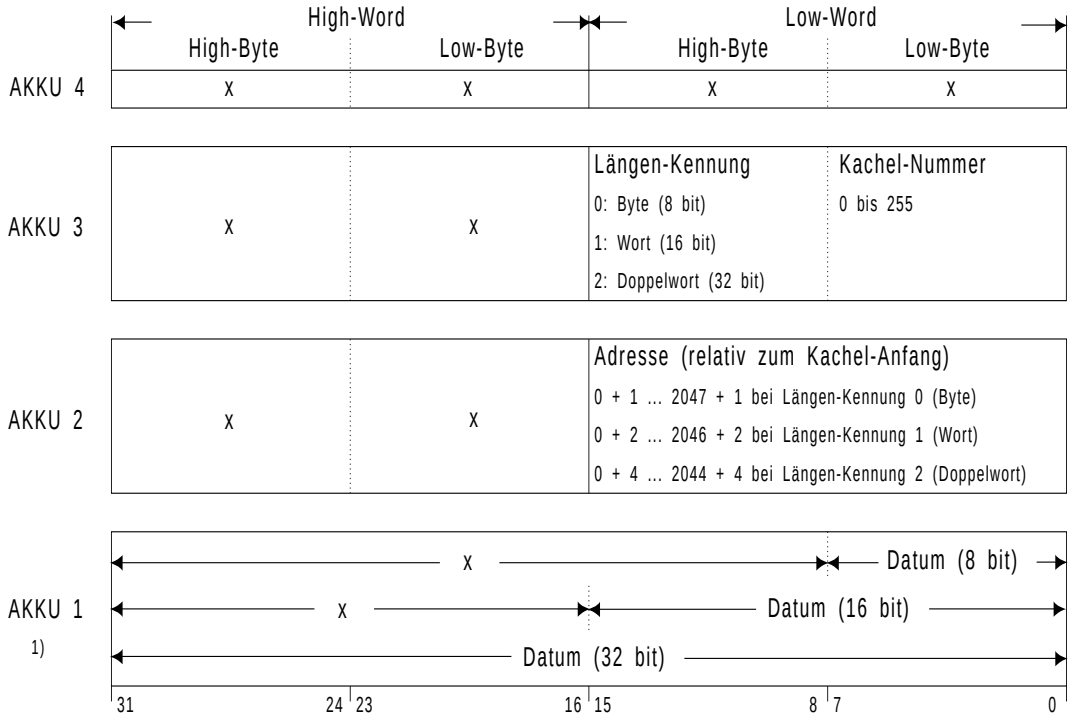
b) AKKU-3-LL

aktuelle Kachel-Nummer,
zulässige Werte: 0 bis 255

c) AKKU-2-L

Quelladresse auf der Kachel,
zulässige Werte: 0 bis 2047

Akku-Belegung **vor** dem Lesen:



1) Datenablage **nach Aufruf** des OB 217

Bild 6-14 AKKU-Belegung vor Aufruf des OB 217

Ergebnis

- Das Lesen der Kachel verläuft **erfolgreich:**
  - **AKKU 1**: enthält (rechtsbündig) den gelesenen Wert (der mögliche Rest der 32 Bits wird gelöscht),
  - **AKKU 3:** wird nicht verändert,
  - **AKKU-2-L:** enthält einen um 1, 2 oder 4 erhöhten Wert (je nach Länge des übertragenen Datums),
  - **VKE:** = 1,
  - **restliche Bit und Wortanzeigen:** werden gelöscht.

- Das Lesen der Kachel ist **nicht möglich**:
  - **alle Akkus:** werden nicht verändert,
  - **VKE:** = 0,
  - **restliche Bit- und Wort-anzeigen:** werden gelöscht.

Fehlerfälle

- falsche Längenkennung in AKKU-3-LH,
- Quelladresse auf der Kachel falsch oder nicht vorhanden,
- angegebene Kchel-Nr. nicht vorhanden.

6.21.3 OB 218: Belegen einer Kachel

Der Sonderfunktions-Organisationsbaustein überträgt die Nummer der CPU zu einer bestimmten Kachel, falls der Inhalt der adressierten Zelle auf dieser Kachel gleich **Null** ist. Solange nun die CPU-Nr. in der Zelle eingetragen bleibt, ist diese Kachel für diese CPU reserviert und kann von anderen CPUs **nicht** belegt werden.
Der Organisationsbaustein OB 218 dient der Synchronisation des Datentransfers und ist besonders wichtig, wenn **größere, zusammengehörige Datenblöcke geschlossen** gesendet bzw. übertragen werden sollen. Im Mehrprozessorbetrieb werden bei jeder Buszuteilung nicht mehr als vier Bytes übertragen. Eine Verriegelung ist daher sinnvoll.

Das **Adressieren** der Kachel, das **Lesen** und das **eventuelle Schreiben** der Steckplatzkennung bilden **eine Programmeinheit**, die nicht unterbrochen werden kann.

Parameter

Akkus

a) AKKU-2-LL

Nummer der zu belegenden Kachel,
zulässige Werte: 0 bis 255

b) AKKU-1-L

Zieladresse auf der Kachel,
zulässige Werte: 0 bis 2047

(Die Inhalte von AKKU 3 und 4 sind irrelevant.)

Akku-Belegung **vor** Aufruf des OB 218:

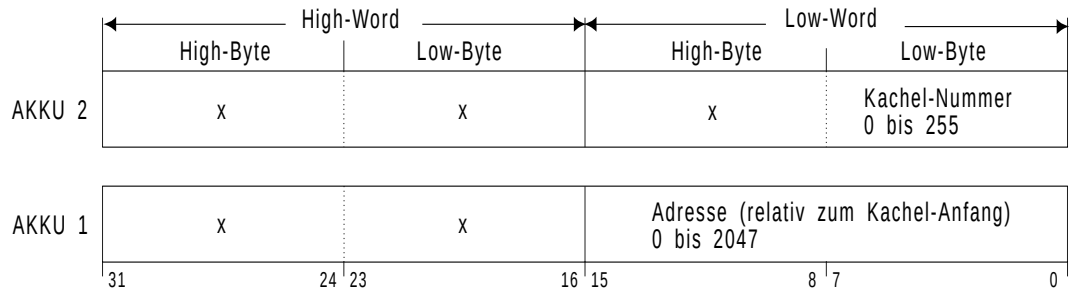


Bild 6-15 Inhalte der AKKUs vor Aufruf des OB 218

Ergebnis

- Das Belegen der Kachel verläuft **erfolgreich:**
  - **alle Akkus:** werden nicht verändert,
  - **VKE:** = 1,
  - **restliche Bit und Wortanzeigen:** werden gelöscht.
- Das Belegen der Kachel ist **nicht möglich**:
  - **alle Akkus:** werden nicht verändert,
  - **VKE:** = 0,
  - **restliche Bit- und Wortanzeigen:** werden gelöscht.

Fehlerfälle

- falsche Längenkennung in AKKU-3-LH,
- Quelladresse auf der Kachel falsch oder nicht vorhanden,
- angegebene Kachel-Nr. nicht vorhanden.

6.21.4
Programmierbeispiel

Aufgabenstellung:

Vom DB 45 einer CPU 928B sollen die Datenwörter 4 bis 11 über den KOR 923C in den DX 45 (Datenwörter 0 bis 7) einer zweiten CPU 928B übertragen werden. Die Synchronisation zwischen Sender und Empfänger (im Mehrprozessorbetrieb) geschieht mittels OB 218.

Aktuelle Kachel auf dem Koordinator: Nr. 255

Koordinierungszelle auf der Kachel (Belegen): Adr. 53

Datenübergabebereich auf der Kachel (Lesen und Schreiben): Adr. 54-69

STEP-5-Operationen im SENDER:

```
        :L    KB    255      Kachelnummer
        :L    KB    53       Adresse Koordinierungszelle
        :SPA  OB    218      Uebertragen Steckplatzkennung in Zelle
        :                    auf Kachel
        :SPB  =M001          Wenn VKE = 1 (Uebertragen erfolgreich),
        :                    Sprung auf Marke
        :BEA                 ansonsten Bausteinende
  M001  :A    DB    45       Quell-Datenbaustein aufschlagen
        :L    KY    2,255    2 = Laengenkennung Doppelwort, Kachelnr.
        :L    KB    54       Anfangsadresse auf Kachel
        :ENT                 Beschreiben von AKKU 3
        :L    DD    4        Datenwoerter 4 und 5 (= 4 byte)
        :SPA  OB    216      Uebertragen 1. Doppelwort
        :                    Adresse um 4 erhoehen (AKKU-2-L = 58)
        :TAK                 Retten der Zieladresse
        :
        :L    DD    6
        :SPA  OB    216      Uebertragen 2. Doppelwort
        :TAK
        :
        :L    DD    8
        :SPA  OB    216      Uebertragen 3. Doppelwort
        :TAK
        :
        :L    DD 10
        :SPA  OB    216      Uebertragen 4. Doppelwort
        :
        :L KY       0,255
        :L KB       53       Adresse mit Steckplatzkennung
        :ENT
        :L KB       0        AKKU 1 = 0
        :SPA  OB    216      Steckplatzkennung loeschen, Datenueber-
        :                    gabebereich freigeben
        :BE
```

Fortsetzung auf der nächsten Seite

Fortsetzung des Beispiels:

STEP-5-Operationen im EMPFÄNGER:

```
        :L    KB    255      Kachelnummer
        :L    KB    53       Koordinierungszelle
        :SPA  OB    218      Kachelbelegung durch 2. CPU
        :SPB  =M002          Wenn VKE = 1, Sprung auf Marke
        :BEA
        :
  M002  :AX   DX    45       Ziel-Datenbaustein
        :L    KY    2,255
        :L    KB    54
        :ENT                 AKKU 3 beschreiben
        :L    KB    0        AKKU 2 beschreiben
        :
        :SPA  OB    217      Lesen 1. Doppelwort
        :                    Adresse um 4 erhoehen (AKKU 2-L = 58)
        :T    DD 0           Transferieren AKKU 1 zu Datenwort 0 und 1
        :
        :SPA  OB    217      Lesen 2. Doppelwort
        :T    DD    2
        :
        :SPA  OB    217      Lesen 3. Doppelwort
        :T    DD    4
        :
        :SPA  OB    217      Lesen 4. Doppelwort
        :T    DD    6
        :
        :L    KY    0,255
        :L    KB    53       Adresse mit Steckplatzkennung
        :ENT
        :L    KB    0        AKKU 1 = 0
        :SPA  OB    216      Steckplatzkennung loeschen, Datenueber-
        :                    gabebereich freigeben
        :BE
```

6.22 OB 220: Vorzeichenerweiterung

Anwendung

Eine Vorzeichenerweiterung ist notwendig, um eine negative 16-bit-Festpunktzahl vor einer Festpunkt-Gleitpunkt-Wandlung (32-bit, Operation FDG) zu einer 32-bit-Festpunktzahl zu erweitern.

Funktion

Diese Sonderfunktion erweitert das Vorzeichen einer 16-bit-Festpunktzahl im AKKU-1-L auf das höherwertige Wort (AKKU-1-H):

- Wenn das Bit $2^{15} = 0$ (positive Zahl) ist, wird das höherwertige Wort mit KH = 0000 geladen.
- Wenn das Bit $2^{15} = 1$ (negative Zahl) ist, wird das höherwertige Wort mit KH = FFFF geladen.

Parameter

AKKU-1-L

16-bit-Festpuktzahl

Ergebnis

AKKU-1-H wird entsprechend dem Vorzeichen der Festpunktzahl in AKKU-1-L geladen (s. o.).

Fehlerfälle

keine

6.23 OB 221: Zyklusüberwachungszeit einstellen

Funktion

Durch Aufruf dieser Sonderfunktion können Sie die Zyklusüberwachungszeit ändern und damit die maximal zulässige Zykluszeit neu festlegen. Standardmäßig ist die Zyklusüberwachungszeit auf 150 ms eingestellt. Mit diesem Aufruf wird gleichzeitig der Timer für die Überwachung neu gestartet: Die maximal zulässige Zykluszeit für den Zyklus, in dem der OB 221 aufgerufen wird, verlängert sich um den neu eingestellten Wert, vom Zeitpunkt des Sonderfunktionsaufrufes an gerechnet. Die Zyklusüberwachungszeit aller folgender Zyklen entspricht dem neu eingestellten Wert (= dem Zeitwert, den Sie im AKKU 1 übergeben).

Parameter

AKKU 1

a) AKKU-1-L

Neue Zykluszeit (in Millisekunden),
zulässige Werte: 1 ms -13000 ms,
positive Festpunktzahl (KF)

b) AKKU-1-H

AKKU-1-H muß den **Wert '0'** haben

Ergebnis

Nach fehlerfreier Bearbeitung des OB 221 ist die neue Zyklusüberwachungszeit eingestellt.

Fehlerfall

Die angegebene Zyklusüberwachungszeit liegt nicht im Bereich 1 ms bis 13000 ms.

Die Funktion wird nicht ausgeführt. Das Systemprogramm erkennt einen Laufzeitfehler und ruft den **OB 31** auf. Die weitere Fehlerreaktion hängt von der Programmierung des OB 31 ab (siehe Abschnitt 5.6.2). Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.

In beiden Fällen wird in AKKU-1-L die Fehlerkennung **1A3AH** hinterlegt.

6.24 OB 222: Zyklusüberwachungszeit neu starten

Funktion

Die Sonderfunktion OB 222 bewirkt ein Nachtriggern der Zyklusüberwachungszeit, d. h. der Timer für die Überwachung wird neu gestartet. Durch Aufruf dieser Sonderfunktion wird die maximal zulässige Zykluszeit für den aktuellen Zyklus um den eingestellten Wert vom Zeitpunkt des Aufrufs an verlängert.

Parameter

keine

Fehlerfälle

keine

6.25 OB 223: Anlaufarten vergleichen

Funktion

Durch Aufruf des OB 223 wird im Mehrprozessorbetrieb überprüft, ob die Anlaufarten **aller** beteiligten CPUs gleich sind.

> **Hinweis**
> Der OB 223 darf **erst dann** aufgerufen werden, wenn **alle CPUs** ihren **Anlauf beendet** haben.
> Bei **ein**geschalteter Anlaufsynchronisation (DX 0) ist dies durch Aufruf des OB 223 im Betriebszustand RUN gewährleistet.
> Bei **ab**geschalteter Anlaufsynchronisation muß dies durch andere Maßnahmen (z. B. verzögerter Aufruf des OB 223) sichergestellt werden.

Parameter

keine

Ergebnis

Fehleranzeigen bei Abweichung der Anlaufarten

Fehlerfälle

Sind die Anlaufarten aller am Mehrprozessorbetrieb beteiligten CPUs nicht gleich, so erkennt diejenige CPU, in der der OB 223 bearbeitet wurde, einen Laufzeitfehler. Es wird der **OB 31** aufgerufen.

Ist der OB 31 nicht geladen, geht die CPU mit der Fehlermeldung LZF in den Stoppzustand. Ihre STOP-LED zeigt langsames Blinken. Die übrigen CPUs gehen mit Dauerlicht in den STOP.

Anzeigen

Bei Aufruf des OB 31 und im Stoppzustand ist in AKKU-1-L die Fehlerkennung **1A3BH** hinterlegt.

6.26 OB 224: Koppelmerker blockweise übertragen

Funktion

Die Übertragung der Koppelmerker erfolgt am Ende jedes Programmzyklus. Im Einzelprozessorbetrieb werden die Koppelmerker jeweils komplett als Datenblock zum Speicher auf dem Koordinator (KOR) oder dem CP und /oder aus diesem Speicher in die Merker der CPU übertragen.

Im Mehrprozessorbetrieb dagegen darf jede CPU nur während der Freigabe ihres Buszugriffes durch den KOR den Bus benutzen. Dadurch wird pro Busfreigabe jeweils nur **ein** Byte übertragen. Anschließend greifen die übrigen CPUs auf den S5-Bus zu. Zusammengehörende Daten, die über mehrere Merkerbytes verteilt sind, werden daher "auseinandergerissen".

Durch Aufruf des Organisationsbausteines OB 224 erreichen Sie eine **blockweise** Übertragung aller im DB 1 der jeweiligen CPU angegebenen Koppelmerker: Solange eine CPU mit der Koppelmerkerübertragung beschäftigt ist, kann sie von einer anderen CPU nicht unterbrochen werden. Da die nächste CPU mit ihrer Übertragung warten muß, wird die zyklische Programmbearbeitung solange verzögert (Zykluszeit!).

Der OB 224 gewährleistet somit eine Datenkonsistenz der gesamten Koppelmerkerinformation. Er muß im Anlaufprogramm aufgerufen werden, und zwar

- in allen am Koppelmerkertransfer beteiligten CPUs

  und

- in jeder verwendeten Anlaufart.

Parameter

keine

Fehlerfälle

keine

6.27 OB 226: Wort aus dem Systemprogramm lesen

Funktion

Das Systemprogramm der CPU hat eine Länge von 128 x 2^{10} Wörtern und liegt in einem Speicherbereich, auf den Sie mit STEP-5-Anweisungen keinen Zugriff haben. Mit Hilfe des OB 226 können Sie jedoch auch aus diesem Speicherbereich einzelne Datenwörter **lesen**.

> **Hinweis**
> Zur Anwendung des OB 226 beachten Sie bitte die Beschreibung des OB 227 und das dazugehörige Programmbeispiel.

Parameter

AKKU 1

Adresse der zu lesenden Systemprogramm-Speicherzelle,.
zulässige Werte: 0 bis 0001 FFFF H

Ergebnis

- **AKKU-1-L**: enthält das gelesene Wort des Systemprogramms
- **AKKU-1-H:** = 0
- **AKKU 2:** enthält den vorherigen Inhalt des AKKU 1 (d. h. die Adresse); der vorherige Inhalt von AKKU 2 geht verloren.

Fehlerfälle

keine

6.28 OB 227: Quersumme des Systemprogramms lesen

Anwendung

Sie können während der zyklischen Programmbearbeitung den Inhalt des Systemprogramms prüfen, indem Sie

- die einzelnen Speicherzellen des Systemprogramms von Adresse 0H bis Adresse 1DFFFH mit Hilfe des OB 226 lesen,
- alle Speicherzelleninhalte mit Festpunktaddition (Befehl +F) aufaddieren, ohne dabei entstehende Überläufe zu beachten,
- mit Hilfe des OB 227 die Quersumme lesen und anschließend
- die durch Festpunktaddition gebildete Endsumme mit der gelesenen Quersumme vergleichen.

Funktion

Der Sonderfunktions-Organisationsbaustein OB 227 lädt die Quersumme des Systemprogramms aus dem Speicherbereich des System in den AKKU 1. Das gelesene Wort entspricht der Summe aller Speicherzellen des Systemprogramms von Adresse 0H bis Adresse 1DFFFH.

Parameter

keine

Ergebnis

- **AKKU 1**: enthält rechtsbündig die gelesene Quersumme (1 Wort); der restliche Inhalt von AKKU 1 ist gelöscht.
- **AKKU 2:** enthält den vorherigen Inhalt des AKKU 1; der vorherige Inhalt von AKKU 2 geht verloren.

Fehlerfälle

keine

Beispiel

Quersumme des Systemprograms überprüfen

Zur Überprüfung der Checksumme des Systemprogramms wird der Funktionsbaustein FB 111 programmiert. Dieser bildet von den Inhalten aller Systemprogram-Speicherwörter die Quersumme und vergleicht diese über den OB 227 mit der im Systemspeicher hinterlegten Quersumme des Systemprogramms. Sind beide Quersummen unterschiedlich, so läuft der FB auf eine STOP-Operation.

MW 250 = Quersumme
MD 252 = Adreßzaehler

```
   FB111
   NAME: CHECKSUM
         :
         :
         :L    KH    0000
         :T    MW    250      Quersummenmerker loeschen
         :T    MD    252      Adresszaehler loeschen
         :
   M001  :SPA  OB    222      Zyklusueberwachungszeit nachtriggern
         :L    MD    252      Adresse der zu lesenden Speicherzelle laden
         :SPA  OB      226              Wort lesen
         :L    MW    250      Quersummenmerker laden
         :+F                  Addieren
         :T    MW    250      Quersummenmerker speichern
         :
         :L    MD    252      Adresszaehler inkrementieren
         :L    KF+1
           :+D                Doppelwort addieren
         :T    MD    252
         :
         :L    DH 0001E000    falls Adresszaehler ungleich '1E000H', ...
         :><   D                ...auf...
         :SPB  =M001            ...Marke M001 springen
         :
         :SPA  OB    227            falls Adresszaehler gleich '1E000H',
         :                    Quersumme lesen
         :L    MW    250      Quersummenmerker laden
         :!=   F              falls gleich, Bausteinende
         :BEB
         :
         :STP                 falls ungleich, Stoppbefehl
         :BE
```

6.29 OB 228: Statusinformation einer Programmbearbeitungsebene lesen

Funktion

Beim Auftreten von bestimmten Ereignissen ruft das Systemprogramm die dazugehörige Programmbearbeitungsebene auf. Die Programmbearbeitungsebene ist damit "aktiviert".
Mit Hilfe des Organisationsbausteins OB 228 können Sie feststellen, ob zu einem Zeitpunkt eine bestimmte Programmbearbeitungsebene aktiviert ist oder nicht. Im AKKU 1 übergeben Sie die Nummer derjenigen Programmbearbeitungsebene, deren Status abgefragt werden soll. (Die Nummern entsprechen den im USTACK unter EBENE eingetragenen Nummern.)

Bei Aufruf des Bausteins hinterlegt dieser die Statusinformation der angegebenen Programmebene im AKKU-1-L. Durch Auswertung dieser Information können Sie Ihre Programmbearbeitung abhängig machen vom Status einer anderen Programmbearbeitungsebene.

Parameter

AKKU-1-L

Nummer der Programmbearbeitungsebene
(siehe USTACK, EBENE),
zulässige Werte (hexadezimal): siehe nachfolgende Tabelle

Ebenen-Nr. in AKKU-1-L	Ebenen-Name
02	NEUSTART
04	ZYKLUS
06	WECKALARM 5 s
08	WECKALARM 2 s
0A	WECKALARM 1 s
0C	WECKALARM 500 ms
0E	WECKALARM 200 ms
10	WECKALARM 100 ms
12	WECKALARM 50 ms
14	WECKALARM 20 ms
16	WECKALARM 10 ms
18	ZEITAUFTRAG
1A	nicht belegt
1C	REGLER-ALARM
1E	nicht belegt
20	nicht belegt
22	nicht belegt
24	PROZESSALARM

Ebenen-Nr. in AKKU-1-L	Ebenen-Name
26	nicht belegt
28	nicht belegt
2A	nicht belegt
2C	Abbruch
2E	Schnittstellenfehler
30	Weckfehler
32	Reglerfehler
34	Zyklusfehler
36	nicht belegt
38	Befehlscodefehler
3A	Laufzeitfehler
3C	Adressierfehler
3E	Quittungsverzug
40	nicht belegt
42	nicht belegt
44	MANUELLER WIEDER-ANLAUF
46	AUTOMATISCHER WIEDERANLAUF

Ergebnis

- **AKKU-1-L**: enthält die Stausinformation:
  = 0 → Programmbearbeitungsebene ist **nicht** aufgerufen
  ≠ 0 → Programmbearbeitungsebene ist aktiviert

- **AKKU-2-L:** enthält den vorherigen Inhalt des AKKU-1-L; der vorherige Inhalt von AKKU-2-L geht verloren.

Fehlerfälle

keine

Beispiel

Ein Quittungsverzug soll im **NEUSTART ignoriert** werden, **nicht jedoch** in den übrigen Programmbearbeitungsebenen.

Sie rufen zu Beginn der Fehlerbearbeitung im OB 23 (QVZ) den OB 228 auf, um festzustellen, ob beim Auftreten des QVZ die Programmbearbeitungsebene NEUSTART (Nummer 02) aktiviert ist oder nicht. Die weitere Fehlerbehandlung machen Sie abhängig von der Statusinformation, die Sie erhalten:

AKKU-1-L = 0: NEUSTART passiv	→	QVZ ist **nicht im NEUSTART**, sondern in einer anderen Programmbearbeitungs-Ebene aufgetreten Das Fehlerprogramm muß bearbeitet werden.
AKKU-1-L ≠ 0: NEUSTART aktiviert	→	QVZ ist im **NEUSTART** aufgetreten, daher darf es ignoriert werden.

Der OB 228 ermöglicht Ihnen somit u. a. eine differenzierte Fehlerbehandlung.

6.30 OB 230 bis 237: Funktionen für Standard-Funktionsbausteine

Die Sonderfunktions-Organisationsbausteine OB 230 bis OB 237 sind für Hantierungsfunktionen reserviert und können nur innerhalb der Standard-Funktionsbausteine FB 120 bis FB 127 aufgerufen werden.

Hantierungsbausterine

Diese Standard-Funktionsbausteine – die sog. "Hantierungsbausteine" – steuern im Einzel- und im Mehrprozessorbetrieb den Datenverkehr über den Kachelbereich: Sie werden eingesetzt, wenn Daten oder Parameter sowie Steuerungsinformationen von den Kommunikationsprozessoren (CPs) übernommen bzw. an die Kommunikationsprozessoren übergeben werden sollen.

Zuordnungshilfe

Der nachfolgenden Tabelle könen Sie entnehmen, von welchen Hantierungsbausteinen die Sonderfunktions-Organisationsbausteine OB 230 bis Ob 237 aufgerufen werden.

Standard-Funktionsbaustein	Sonderfunktions-Organisations-Baustein	Hantierungs-Baustein
FB 120	SF-OB 230	SEND
FB 121	SF-OB 231	RECEIVE
FB 122	SF-OB 232	FETCH
FB 123	SF-OB 233	CONTROL
FB 124	SF-OB 234	RESET
FB 125	SF-OB 235	SYNCHRON
FB 126	SF-OB 236	SEND ALL
FB 127	SF-OB 237	RECEIVE ALL

Anwendung der Hantierungsbausteine

Zur Anwendung der Hantierungsbausteine, die als Software-Produkt auf Diskette zu beziehen sind, gibt es eine detaillierte Betriebsanleitung mit dem Titel "Automatisierungsgerät S5-135U Hantierungsbausteine für R-Prozessor und CPU 928/928B" /8/.

6.31 OB 240 bis 242: Sonderfunktionen für Schieberegister

6.31.1 Schieberegister

In der nachfolgenden Einführung erfahren Sie, wozu Sie Schieberegister anwenden können und was Sie dabei beachten müssen.

Anwendung

Schieberegister können dazu benutzt werden um z. B. in einem Fertigungsbetrieb mit dem Automatisierungsgerät eine Materialverfolgung zu programmieren. Auf der CPU 928B stehen Ihnen maximal 64 Software-Schieberegister zur Verfügung.

Sie können Daten in das Schieberegister schreiben und Daten aus dem Schieberegister lesen. Dies erfolgt über die sog. "Zeiger": Zeiger sind Merkerbytes, die den Inhalt einzelner Zellen eines Schieberegisters enthalten.

Aufbau

Ein Software-Schieberegister besteht aus nebeneinander aufgereihten, 8-bit-breiten Speicherzellen und kann zwischen 2 und maximal 256 Speicherzellen lang sein.

Lage im DB-RAM

Die Daten eines Schieberegisters liegen im Datenbaustein-RAM der CPU. Jedes Schieberegister ist einem bestimmten Datenbaustein fest zugeordnet und hat daher dieselbe Nummer wie der Datenbaustein (zulässig: 192 bis 255). Wenn Sie z. B. ein Schieberegister mit der Nummer 210 eingerichtet haben, so stehen die dazugehörigen Daten im Datenbaustein DB 210.

Das DB-RAM umfaßt ca. 46K byte (Adresse KH 8000 bis KH DD7F). In diesem Bereich liegen die mit OB 254 und 255 kopierten Datenbausteine (von KH 8000 an aufsteigend) und die vom Ihnen eingerichteten Schieberegister (von KH DD7F an abfallend). Falls beim Kopieren von DB oder Einrichten von Schieberegistern der Speicherbereich des DB-RAM nicht ausreicht, erkennt die CPU einen Laufzeitfehler und ruft den OB 31 auf. Die weitere Reaktion hängt von der Programmierung des OB 31 ab (siehe Kapitel 5.6.2).

Die nachfolgenden Abbildungen zeigen das Prinzip eines Software-Schieberegisters mit 3 Zeigern und 12 Speicherzellen:

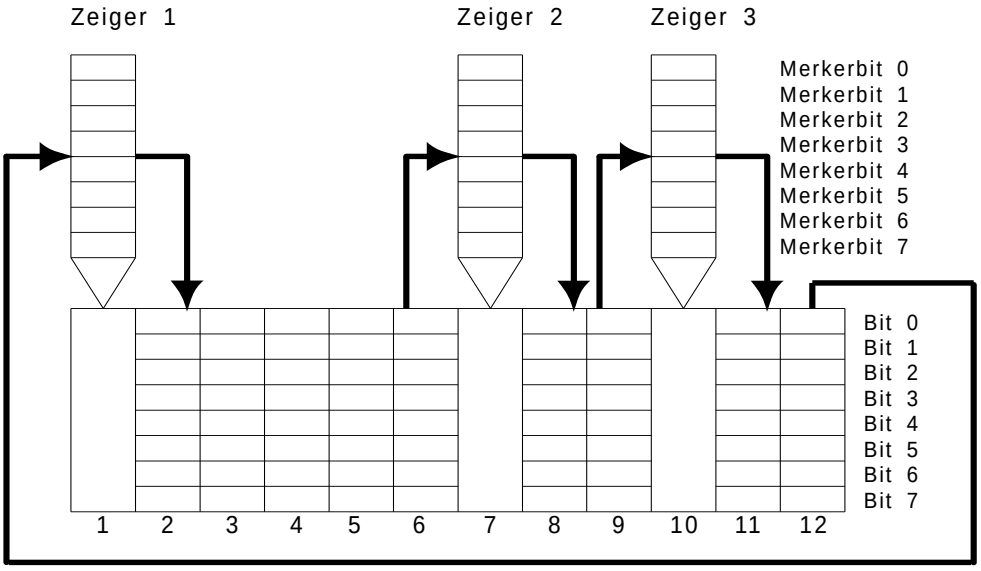


Bild 6-16 Prinzipskizze eines Schieberegisters mit 3 Zeigern und 12 Speicherzellen

Initialisieren

Beim Initialisieren eines Schieberegisters (siehe Abschnitt 6.25.2) geben Sie die Nummer des Merkerbytes für den Zeiger 1 (= Basiszeiger) an. Dieser wird damit fest auf die erste Speicherzelle des Schieberegisters eingestellt. Alle weiteren Zeiger positionieren Sie nun relativ zum Basiszeiger, wobei Sie pro Schieberegister zwischen 1 und maximal 6 Zeiger verwenden können.

Schieben

Beim Schieben eines Schieberegisters wird - wie bei einem Hardware-Schieberegister - der gesamte Inhalt aller Schieberegisterzellen byteweise um eine Position von einer Speicherzelle zur nächsten übertragen (siehe Bild 6-17). Jeder Aufruf der Schieberegisterfunktion bewirkt also ein Verschieben der Information um genau 1 Speicherzelle (entspricht 1 Takt). Die Zeiger werden dabei mit neuen Inhalten versorgt. Entsprechend den eingezeichneten Pfeilen wird die Information durch das gesamte Schieberegister bis in die letzte Speicherzelle "durchgeschoben", von wo aus sie wiederum in die Speicherzelle 1 gelangt. (Beim abgebildeten Schieberegister ist dies nach 12 Takten der Fall.)

Beispiel

Die Bilder 6-17 und 6-18 zeigen das Verschieben der Information innerhalb eines Schieberegisters mit 3 Zeigern und 12 Speicherzellen.

Vor Aufruf der Sonderfunktion werden in den Zeigern (Merkern) bestimmte Bits zur Kennzeichnung der Zeigerinformation gesetzt:

```
   Merkerbit 0 von Zeiger 1 setzen                    :S M 0.0

   Merkerbit 3 von Zeiger 2 setzen                    :S M 1.3

   Merkerbit 2 von Zeiger 3 setzen                    :S M 2.2

Dann wird die Schieberegister-Funktion aufgerufen     :SPA OB 241
```

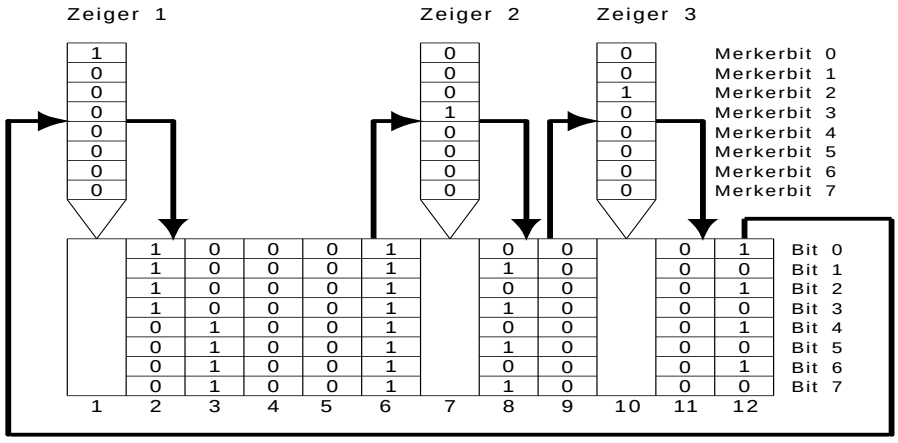


Bild 6-17 Prinzipskizze des Schieberegisters mit 3 Zeigern und 12 Speicherzellen **vor** dem ersten Takt

Nach Aufruf der Sonderfunktion ist 8-bit-breite Information um eine Zelle verschoben:

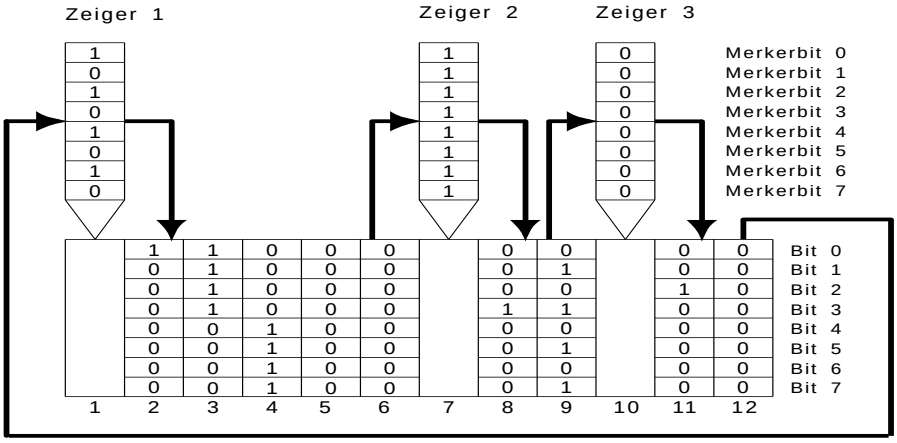


Bild 6-18 Prinzipskizze des Schieberegisters mit 3 Zeigern und 12 Speicherzellen **nach** dem ersten Takt

Fortsetzung auf der nächsten Seite

```
Fortsetzung des Beispiels

Die Information, die sich jetzt in den Zeigern befindet, können Sie auswerten
mit

   :L MB 0
   :
   usw.

Merkerbit 0, 3 und 2 lassen sich am Basiszeiger abfragen: Auf diese Weise kann
die gesamte Information, die aus den Einträgen in allen Zeigern stammt, am
Basiszeiger ausgewertet wer
```

Organisationsbausteine

Wenn Sie ein Schieberegister verwenden wollen, so stehen Ihnen dazu 3 Sonderfunktions-Organisationsbausteine zur Verfügung:

- **OB 240**:

  Diese Funktion **initialisiert** ein Schieberegister.

- **OB 241**:

  Diese Funktion **bearbeitet** ein Schieberegister.

- **OB 242**:

  Diese Funktion **löscht** ein Schieberegister.

6.31.2 OB 240: Schieberegister initialisieren

Anwendung

Vor der Bearbeitung eines Schieberegisters muß dieses zuerst initialisiert werden. Dazu wird einmalig - zweckmäßigerweise in einem Anlauf-Organisationsbaustein - der OB 240 aufgerufen.
Die Parameter, die der OB 240 zum Einrichten eines bestimmten Schieberegisters benötigt, stehen in einem Datenbaustein mit der Nummer des zu initialisierenden Schieberegisters. Zulässig sind damit DB-Nummern zwischen 192 und 255.

Funktion

Mit den Informationen aus dem aufgeschlagenen Datenbaustein wird ein bestimmter Speicherbereich am Ende des DB-RAMs reserviert und initialisiert.

Parameter

aufgeschlagener Datenbaustein

zulässige Werte: DB-Nr. 192 bis 255

Der Datenbaustein ist nach einem festen Schema aufgebaut, das unbedingt eingehalten werden muß. Er kann maximal 9 Datenwörter lang sein (DW 0 bis DW 8).

0	DW 0
Schieberegisterlänge (Bytes) L	DW 1
Nummer des 1. Merkerbytes/Basiszeiger	DW 2
Abstand n_2	DW 3
Abstand n_3	DW 4
Abstand n_4	DW 5
Abstand n_5	DW 6
Abstand n_6	DW 7
0	DW 8 oder letztes Datenwort

Bild 6-19 Aufbau des Datenbausteins zur Initialisierung des Schieberegisters

Die einzelnen Datenwörter müssen folgendermaßen belegt werden:

Datenwort 0

Muß immer den Inhalt 0 haben.

Datenwort 1

Die Schieberegisterlänge L ist die Anzahl (in Bytes) der Speicherzellen des Schieberegisters. Sie kann im Bereich $2 \leq L \leq 256$ liegen.

Datenwort 2

Die Nummer des ersten Merkerbytes legt den Basiszeiger fest und damit den Merkerblock, der den Zeigern zugeordnet wird. Der Merkerblock umfaßt die Gesamtzahl der von Ihnen festgelegten Zeiger. Die Festlegung der weiteren Zeiger geschieht durch Einträge in die Datenwörter DW 3 bis maximal DW 7, wobei pro Zeiger ein Datenwort verwendet wird.
Wenn Sie z.B. zwei weitere Zeiger einrichten wollen, sind dies zusammen mit dem Basiszeiger drei Zeiger.
Achten Sie darauf, daß bis zum Ende des Merkerblocks noch genügend Merker für alle parametrierten Zeiger zur Verfügung stehen.

Datenwort 3 bis maximal 7

Die weiteren Zeiger werden indirekt angegeben: Sie werden durch ihren jeweiligen Abstand (Schieberegister-Zellen = Anzahl Bytes) vom Basiszeiger definiert.

n_2 = Abstand von Zeiger 2 zum Basiszeiger

n_3 = Abstand von Zeiger 3 zum Basiszeiger

n_4 = Abstand von Zeiger 4 zum Basiszeiger

usw. (1 bis maximal 5 Einträge)

letztes Datenwort (DW 4 bis maximal DW 8)

(Im Beispiel DW 8) Muß immer den Inhalt 0 haben.
Werden zum Basiszeiger nur zwei weitere Zeiger parametriert, so steht die ’0’ im Datenwort DW 5 usw..

Alle Angaben erfolgen als Festpunktzahlen.

Hinweis
Die **Anzahl** der Zeiger (maximal 6 inklusive Basiszeiger) darf nicht größer sein als die Länge des Schieberegisters!

Der **Abstand** eines Zeigers zum Basiszeiger darf nicht größer sein als die Länge des Schieberegisters.

Das Datenwort **DW 0** und das **Datenwort nach dem letzten Zeigerabstand** müssen immer den **Inhalt 0** haben.

Der Datenbaustein muß **vor** dem Aufruf des OB 240 **aufgeschlagen** werden!

Der aufgerufene Datenbaustein muß eine **Nummer** aus dem Bereich **DB 192** bis **DB 255** haben.

Speicherplatz

Für jedes Schieberegister werden

n = Schieberegisterlänge/2 + 8 Datenwörter

benötigt, d. h. die Länge des DB-RAMs verringert sich um n Datenwörter, wobei sich die Datenbaustein-RAM-Endadresse zu niedrigeren Adressen verschiebt. Wenn ein Schieberegister, das initialisiert werden soll, schon vorhanden ist, wird bei gleicher Länge des neuen und des bereits vorhandenen Schieberegisters der schon belegte Bereich neu initialisiert. Andernfalls wird der alte Bereich für ungültig erklärt und ein neuer Bereich eröffnet.

Fehlerfälle

- unzulässige Datenbausteinnummer (<192),
- vorhandener Speicherplatz im DB-RAM nicht ausreichend,
- formaler Fehler im Aufbau des Datenbausteins,
- unzulässige Längenangabe für das Schieberegister,
- Parametrierungsfehler bei den Zeigern.

Im Fehlerfall erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf. Die weitere Reaktion hängt von der Programmierung des OB 31 ab (siehe Abschnitt 5.6.2). Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen sind in AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher beschreiben.

6.31.3 OB 241: Schieberegister bearbeiten

Der Sonderfunktions-Organisationsbaustein OB 241 bearbeitet ein Schieberegister, das zuvor durch den OB 240 initialisiert worden sein muß.
In der CPU 928B können maximal 64 Schieberegister aufgerufen werden.

Anwendung

Vor dem Aufruf des OB 241 werden im Normalfall bestimmte Merkerbits in den Zeigern gesetzt/rückgesetzt.
Bei jedem Aufruf des OB 241 wird die Information byteweise von einer Speicherzelle in die nächsthöhere Speicherzelle verschoben. Die Zeiger werden dabei entsprechend mit neuen Inhalten versorgt. Durch wiederholtes Aufrufen des OB 241 kann die Information durch das gesamte Schieberegister bis in die letzte Speicherzelle "durchgeschoben" werden, von wo aus sie wiederum in die Speicherzelle 1 gelangt.

Funktion

Bei jeder Bearbeitung des OB 241 wird das per AKKU-1-L adressierte Schieberegister um eine Position nach rechts verschoben.

Parameter

AKKU-1-L

Nummer des zu bearbeitenden Schieberegisters,
zulässige Werte: 192 bis 255

Ergebnis

Nach dem Aufruf des OB 241 enthalten die Zeiger (maximal 6 pro Schieberegister, mit Ausnahme des Basiszeigers beliebig positionierbar) die Information der davorliegenden Speicherzelle. Diese Information kann dann ausgewertet werden.

Fehlerfälle

- unzulässige Schieberegister-Nummer im AKKU 1,
- Schieberegister nicht initialisiert.

Im Fehlerfall erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf. Die weitere Reaktion hängt von der Programmierung des OB 31 ab (siehe Abschnitt 5.6.2). Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen sind in AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher beschreiben.

6.31.4 OB 242: Schieberegister löschen

Funktion

Mit dieser Funktion wird ein Schieberegister im Datenbaustein-RAM "gelöscht": der Eintrag in der Adressliste DB 0 wird gelöscht und das entsprechende Schieberegister im DB-RAM für ungültig erklärt (Achtung: Auch gelöschte Schieberegister belegen weiterhin Speicherplatz!).

Parameter

AKKU-1-L

Nummer des zu löschenden Schieberegisters,
zulässige Werte: 192 bis 255

Ergebnis

Nach Aufruf des OB 242 ist das Schieberegister gelöscht und kann nicht mehr verwendet werden; soll es wieder bearbeitet werden, muß es erneut initialisiert werden.

Fehlerfälle

- unzulässige Schieberegister-Nummer im AKKU 1,
- Schieberegister nicht initialisiert .

Im Fehlerfall erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf. Die weitere Reaktion hängt von der Programmierung des OB 31 ab (siehe Abschnitt 5.6.2). Ist der OB 31 nicht geladen, geht die CPU in den Stoppzustand.
In beiden Fällen sind in AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher beschreiben.

6.32 OB 250/251: Regelung/ PID-Algorithmus

In der CPU 928B des AG 135U können Sie einen oder mehrere PID-Regler aufrufen. Jeder Regler muß im Anlauf-Organisationsbaustein initialisiert werden. Zur Übergabe von Parametern wird ein Datenbaustein verwendet.
Der eigentliche Regel-Algorithmus ist im Systemprogramm integriert und vom Anwender lediglich als Organisationsbaustein aufrufbar. Als Datenschnittstelle zwischen Regel-Algorithmus und dem Anwenderprogramm dient ein Datenbaustein.

6.32.1 Funktionsbeschreibung des PID-Reglers

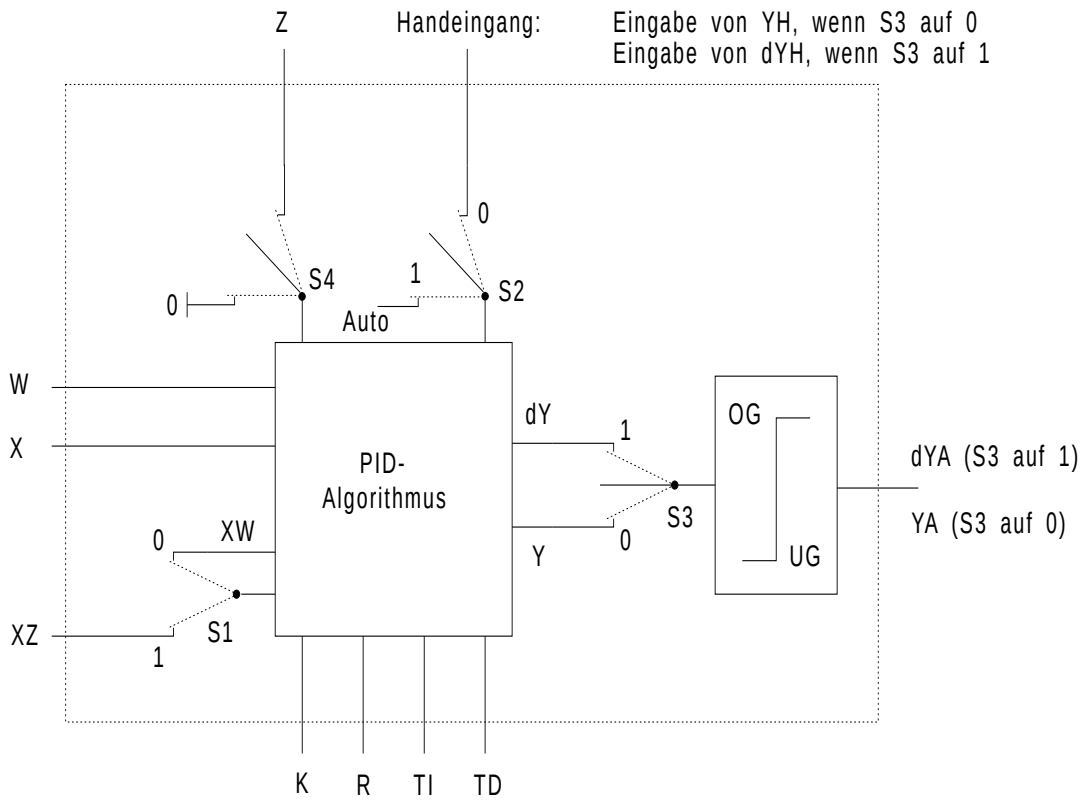


Bild 6-20 Blockschaltbild des PID-Reglers

Index k **k-te Abtastung**

Schalter	Stellung	Wirkung
S1 STEU-BIT 1	0 1	Dem Differenzierer wird die Regeldifferenz XW_k zugeführt. Dem Differenzierer kann über XZ ein anderes Signal zugeführt werden.
S2 STEU-BIT 0	0 1	Handbetrieb Automatik
S3 STEU-BIT 3	0 1	Stellungs-Algorithmus Geschwindigkeits-Algorithmus
S4 STEU-BIT 5	0 1	Mit Störgrößenaufschaltung Ohne Störgrößenaufschaltung

Steuerwort STEU

Eine den Schalterstellungen dieses Blockschaltbildes entsprechende Funktion wird bei der Parametrierung des PID-Reglers dadurch erzielt, daß im Steuerwort STEU die Steuerbits passend gesetzt werden. Der kontinuierliche Regler ist für schnelle Regelstrecken, wie sie z.B. in der Verfahrenstechnik als Druck-, Temperatur- oder Durchflußregelungen auftreten, ausgelegt.

PID-Alorithmus

Dem Regler selbst liegt ein PID-Algorithmus zugrunde. Sein Ausgangssignal kann wahlweise als Stellgröße (Stellungs-Algorithmus) oder als Stellgrößenänderung (Geschwindigkeits-Algorithmus) ausgegeben werden.
Die einzelnen P-, I- und D-Anteile sind über ihre jeweiligen Parameter R, TI und TD abschaltbar, indem die betreffenden Zellen mit Null vorbesetzt werden. Damit können alle gewünschten Reglerstrukturen, z.B. PI-, PID- oder PD-Regler, leicht realisiert werden.

Differenzierer

Dem Differenzierer kann wahlweise die Regeldifferenz XW oder - über den XZ-Eingang- eine beliebige Störgröße oder der invertierte Istwert -x zugeführt werden.

Störkompensation

Für den Fall, daß zur Kompensation eines Störgrößeneinflusses eine Vorsteuerung des Stellgliedes ohne Zeitverhalten erwünscht ist, kann dem Regel-Algorithmus eine im Prozeß meßtechnisch erfaßbare Störgröße Z aufgeschaltet werden. Im Handbetrieb wird an dieser Stelle die vorgewählte Stellgröße YN übernommen.

Invertierter Reglersinn

Wenn ein invertierter Reglersinn gefordert wird, ist ein negativer K-Wert vorzugeben.

Begrenzung der Stellinormation

Wenn die Stellinformation (dY oder Y) an einer Begrenzung liegt, wird der I-Anteil automatisch abgeschaltet, um eine Verschlechterung des Reglerverhaltens zu vermeiden.

Das Reglerprogramm kann durch Vorgabe von Festwerten oder über adaptive (dynamische) Vorgabe von Parametern (K, R, TI, TD) versorgt werden. Die Eingabe erfolgt über die den einzelnen Parametern zugeordneten Speicherzellen.

6.32.2 PID-Algorithmus

Dem PID-Regler liegt ein Geschwindigkeits-Algorithmus zugrunde, nach dem zu einem bestimmten Zeitpunkt t = k · TA das jeweilige Stellinkrement dY_k nach folgender Formel berechnet wird:

$$dY_k = K \left[(XW_k - XW_{k-1})\, R + \frac{TA}{2TN} (XW_k + XW_{k-1}) + \frac{1}{2} \left\{ \frac{TV}{TA} (XU_k - 2XU_{k-1} + XU_{k-2}) + dD_{k-1} \right\} \right]$$

$$= K (\quad dPW_k R \quad + \quad dI_k \quad + \quad dD_k \quad)$$

$dXXX_k$: Änderung der Größe XXX zum Zeitpunkt t.

U kann W oder Z sein, je nachdem, ob dem Differenzierer XW oder XZ zugeführt wird. Entsprechend gilt:

Bei XW_k-Zuführung:	Bei XZ-Zuführung:
$PW_k = W_k - X_k$	
$PW_k = XW_k - XW_{k-1}$	$PZ_k = XZ_k - XZ_{k-1}$
$QW_k = PW_k - PW_{k-1}$	$QZ_k = PZ_k - PZ_{k-1}$
$QW_k = XW_k - 2XW_{k-1} + XW_{k-2}$	$QZ_k = XZ_k - 2XZ_{k-1} + XZ_{k-2}$

$$dPW_k = (XW_k - XW_{k-1})R$$

$$dI_k = TI \cdot XW_k \qquad TI = \frac{TA}{TN}$$

$$dD_k = \frac{1}{2}(TD \cdot QU_k + dD_{k-1}) \qquad TD = \frac{TV}{TA}$$

Wenn als Reglerausgang zum Zeitpunkt t_k die Stellgröße Y_k gewünscht wird, wird sie nach folgender Formel gebildet:

$$Y_k = \sum_{m=o}^{m=k} dY_m$$

Bei den meisten Reglerentwurfsverfahren geht man davon aus, daß R = 1 ist, wenn ein P-Verhalten erwünscht ist.

Mit der Größe R kann der Proportionalanteil des PID-Reglers eingestellt werden.

Datenbausteine für den PID-Regler

Die reglerspezifischen Daten werden mit Hilfe eines Übergabe-Datenbausteins eingegeben (Initialisierung und Bearbeitung des PID-Reglers siehe Kapitel 6.11.1 und 6.11.2).

Diese Daten müssen Sie im Übergabe-Datenbaustein x vorgeben:

K, R, TI, TD, W, STEU, YH, BGOG, BGUG

Der Übergabe-Datenbaustein muß aus 49 Datenwörtern mit den Nummern 0 bis 48 bestehen. Die Datenbelegung dieser Datenwörter wird in der nachfolgenden Tabelle erläutert.

Aufbau des Übergabe-Datenbausteins

Tabelle 6-10 Übergabe-Datenbaustein für PID-Regelung

Adr. im DB	Name	E/A 1)	Zahlen-format 2)	PG-Format 3)	Bemerkung
DW 0	–	–	–	–	Reserve
DD 1	K	E	GP	KG	Proportionalbeiwert K >0: Positiver Regelsinn d.h. gleichsinnige Änderung von Sollwert und Stellgröße K <0: Negativer Regelsinn; Gleitpunktzahlen-bereich
DD 3	R	E	GP	KG	R-Parameter, üblicherweise = 1 bei Reglern mit P-Anteil
DD 5	TI	E	GP	KG	TI = TA/TN
DD 7	TD	E	GP	KG	TD = TV/TA
DD 9	W_k	E	GP	KG	Sollwert-Eingabe hier, wenn STEU-Bit 6 = 1, ansonsten in Wort Nr. 19 ($-1 \leq W_k < 1$)
DW 11	STEU	E	BM	KM	Steuerwort
DD 12	YH_k	E	GP	KG	Handwert-Eingabe hier, wenn STEU-Bit 6 = 1; ansonsten in Wort Nr. 18 ($-1 \leq YH_k < 1$) Bei Geschwindigkeits-Algorithmus sind hier Stellwert-Inkremente anzugeben
DD 14	BGOG	E	GP	KG	Oberer Begrenzungswert 4) $-1 \leq BGOG \leq 1$ ($YA_{k\,max}$); !! BGUG < BGOG !!
DD 16	BGUG	E	GP	KG	Unterer Begrenzungswert 4) $-1 \leq BGUG \leq 1$ ($YA_{k\,min}$)
DW 18	YH_k	E	LP	KF	Handwert-Eingabe hier, wenn STEU-Bit 6 = 0 ($-1 \leq YH < 1$) Bei Geschwindigkeits-Algorithmus sind hier Stellwert-Inkremente anzugeben.
DW 19	W_k	E	LP	KF	Sollwert-Eingabe hier, wenn STEU-Bit 6 = 0 ($-1 \leq Wk < 1$)
DW 20	MERK	–	BM	KF	Bit 0 = 1: positive Begrenzung überschritten; Bit 1 = 1: negative Begrenzung unterschritten
DW 21	X_k	E	LP	KF	Istwert-Eingabe für STEU-Bit 7 = 0 ($-1 \leq X_k < 1$)
DD 22	X_k	E	GP	KG	Istwert-Eingabe für STEU-Bit 7 = 1 ($-1 \leq X_k < 1$)

Adr. im DB	Name	E/A 1)	Zahlen-format 2)	PG-Format 3)	Bemerkung
Fortsetzung der Tabelle 6-10:					
DW 24	Z_k	E	LP	KF	Störgröße ($-1 \leq Z_k < 1$)
DD 25	Z_k	E	GP	KG	Störgrößeneingabe hier, wenn STEU-Bit 7 = 1 ($-1 \leq Z_k < 1$)
DD 27	Z_{k-1}	–	GP	KG	Vergangenheitswert der Störgröße
DW 29	XZ_k	E	LP	KF	Über den Eingang XZ dem Differenzierer zugeführte Größe ($-1 \leq XZ_k < 1$); Eingabe hier, wenn STEU-Bit 7= 0
DD 30	XZ_k	E	GP	KG	XZ-Eingabe hier, wenn STEU-Bit 7 = 1 ($-1 \leq XZ_k < 1$)
DD 32	XZ_{k-1}	–	GP	KG	Vergangenheitswert von XZ_k
DD 34	PZ_{k-1}	–	GP	KG	XZ_{k-1} - XZ_{k-2}
DD 36	dD_{k-1}	–	GP	KG	Differentialanteil
DD 38	XW_{k-1}	–	GP	KG	Vergangenheitswert der Regeldifferenz
DD 40	PW_{k-1}	–	GP	KG	XWk-1 - XW_{k-2}
DW 42	–	–	–	–	Reserve
DD 44	Y_{k-1}	–	GP	KG	Vergangenheitswert der berechneten Stellgröße Yk-1 bzw. dY_{k-1} vor dem Begrenzer
D 46	YA_k	A	GP	KG	Ausgangsgröße
DW 48	YA_k	A	LP	KF	Ausgangsgröße BGUG ≤ YA ≤ BGOG

1) E = Eingabe, A = Ausgabe

2) GP = Gleitpunktzahl, LP = Linkspunktzahl (siehe Seite 6 - 103)

3) Vorgeschlagenes Format (KH, KM ebenfalls zulässig)

4) Im Linkspunktformat müssen oberer und unterer Begrenzungswert nach folgenden Formeln eingegeben werden:

DD 14 = BGOG:

$$\text{Wert als Gleitpunktzahl} = \frac{\text{BGOG}}{32767}$$

DD 16 = BGUG:

$$\text{Wert als Gleitpunktzahl} = \frac{\text{BGUG}}{32767}$$

Beispiel für Begrenzungswerte

- **Begrenzungswerte**

  Oberer Begrenzungswert = 0,1

  Unterer Begrenzungswert = -0,1

- Einträge in den DB:

  DD 14: +1000 000 +00

  DD 16: -1000 000 +00

- Ausgangsgröße wird begrenzt:

  DW 48: ±3276

  DD 15: ±0,1

Anmwerkung:

Für Begrenzungswerte außerhalb 1 wird die Ausgangsgröße im GP-Format begrenzt (DD 46).

Belegung des Steuerwortes STEU (Datenwort DW 11 im Übergabe-DB)

Tabelle 6-11 Steuerwort im Übergabe-DB

DW 11 Bit-Nr.	Name	Bedeutung
11.0	AUTO	= 1: Automatikbetrieb = 0: Handbetrieb
11.1	XZ_EIN	= 1: Dem Differenzierer wird über den XZ-Eingang eine andere Größe zugeführt,die nicht XW_k sein darf. = 0: Dem Differenzierer wird XW_k zugeführt. Der XZ-Eingang bleibt unberücksichtigt.
11.2	REG_AUS	= 1: Beim Aufruf des Reglers (OB 251) werden mit Ausnahme von K, R, TI, TD, BGOG, BGUG, STEU, YH_k, W_k, Z_k und Z_{k-1} alle anderen Größen (DW 20 bis DW 48) im Regler-DB einmal gelöscht. Der Regler ist ausgeschaltet. Der Vergangenheitswert der Störgröße wird aktualisiert. = 0: Regeln
11.3	GESCHW	= 1: Geschwindigkeits-Algorithmus = 0: Stellungs-Algorithmus
11.4 [1)]	HANDART	= 1: Bei GESCHW = 0 (Stellungs-Algorithmus) wird die zuletzt ausgegebene Stellgröße beibehalten. Bei GESCHW = 1 (Geschwindigkeits-Algorithmus) wird das Stellinkrement $dY_k = 0$ gesetzt. = 0: Bei GESCHW = 0 wird nach dem Umschalten auf Handbetrieb der ausgegebene Stellwert YA in 4 Abtastschritten exponentiell auf den eingestellten Handwert geführt. Danach werden weitere Handwerte sofort am Reglerausgang übernommen. Bei GESCHW = 1 werden die Handwerte sofort auf den Reglerausgang durchgeschaltet. Im Handbetrieb sind die Begrenzungen wirksam. Im Handbetrieb werden folgende Größen aktualisiert: X_k, XW_{k-1} und PW_{k-1} XZ_k, XZ_{k-1} und PZ_{k-1}, wenn STEU-Bit 1 = 1 Z_k und Z_{k-1}, wenn STEU-Bit 5 = 0 Die Größe dD_{k-1} wird = 0 gesetzt. Der Algorithmus wird nicht berechnet.
11.5	NO_Z	= 1: keine Störgrößenaufschaltung = 0: mit Störgrößenaufschaltung
11.6	PGDG	= 1: W_k-, YH_k-Eingabe als Gleitpunktzahl = 0: Eingabe als Linkspunktzahl

DW 11 Bit-Nr.	Name	Bedeutung
Fortsetzung der Tabelle 6-11:		
11.7	VAR_GP	= 1: Die Variablen X_k, XZ_k und Z_k werden als Gleitpunktzahl eingegeben. = 0: Eingabe der Variablen als Linkspunktzahl
11.8	STOS	= 1: keine stoßfreie Hand-Automatik-Umschaltung = 0: stoßfreie Hand-Automatik-Umschaltung
11.9 bis 11.15		ohne Bedeutung

1) Nur bei Handbetrieb (AUTO = 0) relevant.

6.32.3 OB 250: PID-Algorithmus initialisieren

Funktion

Der OB 250 initialisiert den PID-Algorithmus und wird in den Anlauf-OBs 20/21/22 aufgerufen.

Parameter

Die für das Initialisieren erforderlichen Parameter stehen im Übergabe-Datenbaustein (DB x).

> **Hinweis**
> Der Übergabe-Datenbaustein muß vor Aufruf des OB 250 aufgeschlagen werden.

Für jeden Regler muß zur Datenübergabe ein eigener DB x verwendet werden ($x \leq 254$). Das Systemprogramm erzeugt daraus automatisch einen weiteren DB x + 1 im Datenbaustein-RAM, den der Regler im zyklischen Betrieb als Datenfeld verwendet; die entsprechenden DB-Nummern müssen also noch frei sein. Die Datenbausteine DB x + 1 sind die jeweilige Datenschnittstelle zwischen den Reglern und dem Anwender bzw. der Peripherie.

Fehlerfälle

Der OB 250 verwendet intern den OB 254 bzw. OB 255 (Duplizieren von Datenbausteinen). Im Fehlerfall erkennt die CPU einen Laufzeitfehler und ruft den OB 31 auf. Wenn dieser nicht programmiert ist, geht die CPU in den Stoppzustand. Die im AKKU 1 hinterlegten Fehlerkennungen beziehen sich dann auf den OB 250.

Hinweis
Wenn bei der Initialisierung der DB x + 1 nicht freigehalten war, wird dieser ohne Meldung vom Systemprogramm als Reglerdatenfeld verwendet, sofern er die gleiche Länge hat wie ein Regler-DB (48 Datenwörter); dabei werden die Datenwörter 20 bis 48 gelöscht. Ansonsten geht die CPU in den Stoppzustand.

Statt Datenbausteinen DB können auch erweiterte Datenbausteine DX verwendet werden. Die Initialisierung verläuft dabei analog zu der bei Datenbausteinen DB.

6.32.4 OB 251: PID-Algorithmus bearbeiten

Anwendung

Der OB 251 wird während der zyklischen Programmbearbeitung aufgerufen und bearbeitet den PID-Algorithmus.

Aufruf

Nach Ablauf der Abtastzeit soll der Regler aufgerufen werden. Halten Sie dabei folgende Reihenfolge ein:

Schritt	Aktion
1	Datenbaustein DB x + 1 aufschlagen
2	Eingangsdaten X_k, XZ_k, Z_k und YH_k oder eine Untermenge davon laden
3	Eingangsdaten formrichtig umwandeln und in den DB x + 1 tranferieren
4	OB 251 (PID-Regler bearbeiten) aufrufen
5	Ausgangsdatums YA_k aus dem DB x + 1 laden
6	Datum umwandeln und zur Prozeßperipherie transferieren

Format der Reglereingänge und -ausgänge

Der PID-Regelalgorithmus verwendet intern zur Zahlendarstellung das Gleitpunktformat und kann mit Gleitpunktwerten versorgt werden. Eine Versorgung des PID-Regelalgorithmus im Linkspunktformat ist ebenfalls möglich (siehe dazu Bit 6 und 7 im Steuerwort STEU). In diesem Fall wandelt der Regler bei jedem Aufruf selbständig die Wörter ins Gleitpunktformat um.

Die Anpassung der Wörter von den Eingabe- und Ausgabebaugruppen im STEP-5-Programm ist laufzeitgünstiger bei Verwendung des Linkspunktformates (siehe Tabelle am Ende dieses Kapitels).

Eingänge

W, YH, X, Z und XZ können wahlweise als Gleit- oder Linkspunktzahl eingegeben werden. Im Datenübergabebaustein sind für jede Größe jeweils unterschiedliche Speicherplätze vorgesehen.

Eingabe als Linkspunktzahl

(Erläuterungen zur Linkspunktzahl: Siehe Tabelle am Ende dieses Kapitels)

Hinweis
Bei Einhaltung der Eingangsnennbereiche der Analogeingabebaugruppen müssen Sie berücksichtigen, daß das Bitmuster für einen bestimmten Eingangswert anders ist als bei Ausnutzung des vollen Eingangsbereichs. Dies ist insbesondere bei der Sollwerteinstellung sehr wichtig, da es sonst vorkommen kann, daß ein über das PG eingegebener Sollwert nicht erreicht werden kann, obwohl der Istwert weit über dem gewünschten Wert liegt.

Wenn der eingesetzte Analog-Digital-Umsetzer die negativen Zahlen als Betrag und Vorzeichen liefert, muß man daraus, bevor sie in den Regler-DB transferiert werden, das Zweierkomplement bilden. Anschließend muß die Binärstelle 15 = 1 gesetzt werden.
Wenn die Zahl -0 als Betrag und Vorzeichen in der Form

1000000000000000

beim verwendeten Analog-Digital-Umsetzer möglich ist, darf davon **kein** Zweierkomplement gebildet werden, sondern die Zahl muß als +0 in den Regler-DB gelangen:

0000000000000000

Ausgang

Der Reglerausgang YA liegt im DB als Links- und als Gleitpunktzahl vor. Linkspunkteingänge und -ausgänge müssen unter Berücksichtigung der verwendeten Ein- und Ausgabebaugruppen (Analog-Digital-Umsetzer, Digital-Analog-Umsetzer) vor und nach dem Regleraufruf im STEP-5-Anwenderprogramm formatgewandelt werden, bevor sie in den bzw. aus dem Regler-DB transferiert werden.

Allgemeine Hinweise

Verwendung von STOS

Wenn STOS (STEU-Bit 8) auf Null steht, verläuft die Umschaltung von Hand- auf Automatikbetrieb stoßfrei; d. h. eine anstehende, beliebig große Regeldifferenz wird nur über den I-Anteil ausgeregelt. Wenn jedoch TI = TA/TN = 0 gewählt wird (P- oder PD-Regler), verursacht die Regeldifferenz bei der Umschaltung keine Änderung der Stellgröße.

Dies kann vermieden werden, indem man STOS = 1 setzt. Eine Regeldifferenz wird dann bei Hand-Automatik-Umschaltung schnell ausgeregelt, gleichgültig ob TI = 0 ist oder nicht. Der dabei entstehende Stellgrößensprung entspricht der Größe der Regeldifferenz, ist also nicht willkürlich im Sinn einer Störung des Reglerbetriebs.

Anzeige von MERK, Bit 0 und 1

Bit 0 und 1 von MERK können, falls gewünscht, zur Anzeige gebracht werden, um anzuzeigen, daß die Stellgröße (bei Geschwindigkeitsalgorithmus das Stellinkrement) in der oberen oder unteren Begrenzung ist. Da diese Bits vom Algorithmus zur Abschaltung des I-Anteils ausgewertet werden, dürfen sie nicht überschrieben werden.

Hinweis
Regler-Datenbausteine DB x+ 1 dürfen während des **zyklischen Betriebs nicht neu geladen werden.**

Kaskadenregelung

Wenn mit zwei oder mehr Reglern eine Kaskadenregelung aufgebaut wird, ist folgendes zu beachten:

- Wenn die Kaskade aufgetrennt wird, müssen entweder alle Regler gleichzeitig in den Handbetrieb gehen, damit kein Regler infolge seines I-Anteils abdriften kann, oder es muß zumindest der Regler des äußeren Kreises im Handbetrieb arbeiten, damit die letzte Stellgröße, die dem Sollwert des inneren Kreises entspricht, beibehalten oder auf einen Sicherheitswert gefahren werden kann.

- Wenn die Kaskade geschlossen werden soll, sollten beide Kreise gleichzeitig oder wenigstens der innere Kreis im Automatikbetrieb arbeiten, damit die Stellgröße des äußeren Kreises als Sollwert übernommen werden kann.

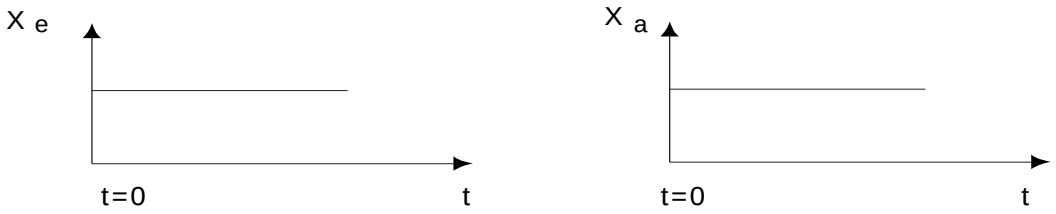
Umschalten auf Handbetrieb

Wenn beim Umschalten auf Handbetrieb die Regelstrecke vom Regler abgetrennt und am Stellglied direkt verstellt wird, ist die so gewonnene Stellgröße über den Handeingang dem Regler zuzuführen. Dies bewirkt, daß beim Umschalten von Hand- auf Automatikbetrieb der Reglerausgang mit der im Handbetrieb eingestellten Stellgröße übereinstimmt. Beim Geschwindigkeits-Algorithmus handelt es sich um die Stellgrößenänderung.

Reglerkenngrößen

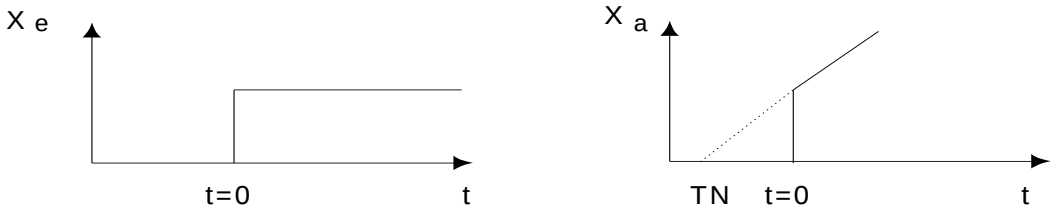
P-Regler

Die Kenngröße für einen P-Regler ist K. Sie ist der Quotient aus Ausgangs- und Eingangsgröße: K = Xa/Xe.



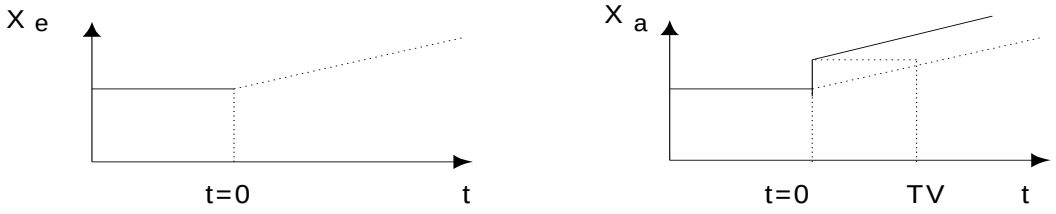
PI-Regler

Die Kenngrößen für einen PI-Regler sind der Proportionalbeiwert K und die Nachstellzeit TN. Der Proportionalbeiwert K ist der Quotient aus Ausgangsgröße und Eingangsgröße und bestimmt das P-Verhalten. Die Nachstellzeit TN ist diejenige Zeit, die bei der Antwort benötigt wird, um infolge der I-Wirkung eine gleichgroße Stellgrößenänderung zu erzielen, wie sie infolge des P-Anteils entsteht.



PD-Regler

Die Kenngröße für einen PD-Regler sind der Proportionalbeiwert K (siehe oben) und die Vorhaltezeit TV. Die Vorhaltezeit ist diejenige Zeit, die ein P-Regler bei konstanter Änderungsgeschwindigkeit der Eingangsgröße benötigen würde, um die gleiche Änderung der Ausgangsgröße zu bewirken, die ein PD-Regler infolge seines D-Anteils sofort bewirkt. Um die Vorhaltezeit TV zu bestimmen, geht man nicht von einer Sprungfunktion aus, sondern von einer linearen Änderung der Eingangsgröße.



PID-Regler

Die Kenngröße für einen PID-Regler sind der Proportionalbeiwert K, die Nachstellzeit TN und die Vorhaltezeit TV. Sie bestimmen jeweils das P-, I- und D-Verhalten.

Parameteränderung

Der P-Anteil der Stellgröße wird nach folgender Formel gebildet:

$$\text{P-Anteil} = KP \cdot (XW_k - XW_{k-1})$$

Wird KP oder R im Automatikbetrieb geändert, wirkt sich dies nur auf nachfolgende Änderungen der Regeldifferenz XW_k aus. Der momentane Wert der Stellgröße bleibt von der Parameteränderung unbeeinflußt. Dieses Verhalten ermöglicht eine stoßfreie Parameteränderung.
Ist dieses Verhalten jedoch nicht erwünscht, so kann es durch folgende Berechnung, die nur einmal bei jeder Parameteränderung vorzunehmen ist, beseitigt werden (Beispiel für KP-Änderung):

$$Y_{k-1} = Y_{k-1} + XW_{k-1}(KP_{neu} - KP_{alt})$$

Wird bei einer Parameteränderung folgendes Programm verwendet, verhält sich der Regler wie ein analoger Regler:

:L	KP_{neu}	KP_{neu} laden
:L	KP_{alt}	KP_{alt} laden
:-G		
:L	DD38	XW_{k-1}
:xG		
:L	DD44	Y_{k-1}
:+G		
:T	DD44	$= Y_{k-1}$

Abkürzungen für PID-Regler

dY_k	Berechnetes Stellinkrement
dZ_k	Störinkrement
GP	Gleitpunktdarstellung
k	k-te Abtastung
K	Proportionalbeiwert
LP	Linkspunktdarstellung
OG	Obere Grenze (Begrenzer)
R	R-Parameter
TA	Abtastzeit
TD	TV/TA
TI	TA/TN
t	Abtastzeitpunkt = k · TA
TN	Nachstellzeit
TV	Vorhaltzeit
UG	Untere Grenze (Begrenzer)
W_k	Sollwert
X_k	Istwert
XW_k	Regeldifferenz
Y_k	Berechnete Stellgröße
YA_k	Stellwert (Stellinkrement oder Stellgröße)
Z_k	Störgröße

Linkspunktzahl

Zur Darstellung einer Linkspunktzahl im Datenbaustein wird ein Wort benötigt. Die Zuordnung zwischen dezimal dargestellter Linkspunktzahl, dual dargestellter Linkspunktzahl und der Darstellung im Format KF am Programmiergerät ist im folgenden Beispiel dargestellt.

Tabelle 6-12 Linkspunktzahlen

Linkspunktzahl in		**Festpunktzahl**
Dezimal-darstellung	**Dualdarstellung**	
-0.999... .	1000000000000001	-32767
-0.75	1010000000000000	-24576
-0.5	1100000000000000	-16384
-0.25	1110000000000000	-8192
0	0000000000000000	0
+0.25	0010000000000000	+ 8192
+0.5	0100000000000000	+16384
+0.75	0110000000000000	+24576
+0.999... .	0111111111111111	+32767

Negative Linkspunktzahlen ergeben sich in der Dualdarstellung durch Zweierkomplementbildung aus positiven Linkspunktzahlen.

Linkspunktzahlen (LF) lassen sich nach der folgenden Beziehung in die am Programmiergerät dargestellten Werte (KF) umrechnen :

$$LP \cdot 32767 = KF$$

mit $-1 < LP < +1$ und $-32767 \leq KF \leq +32767$

6.33 OB 254/255: Einen Datenbaustein verschieben/duplizieren

Mit den Sonderfunktionen OB 254/255 übertragen Sie einzelne Datenbausteine vom Anwenderspeicher in das DB-RAM(Datenbausteinspeicher) der CPU. Die Sonderfunktionen OB 254 und OB 255 laufen identisch ab, wobei der OB 254 ausschließlich für DX-Datenbausteine und der OB 255 für DB-Datenbausteine zuständig ist.

Anwendung

Verschieben oder Duplizieren eines Datenbausteins.

Funktion

Verschieben

Ein Datenbaustein wird vom Anwenderspeicher in das DB-RAM **verschoben**.

Ein Datenbaustein im Anwenderspeicher wird unter **Beibehaltung seiner ursprünglichen Bausteinnummer** in das DB-RAM verschoben. Die neue Anfangsadresse wird in die Adreßliste im DB 0 eingetragen.

Duplizieren

Ein Datenbaustein im Anwenderspeicher oder im DB-RAM wird mit einer **neuen Baustein-Nr.** ins DB-RAM kopiert. Die Anfangsadresse des neuen Datenbausteins wird in die Adreßliste im DB 0 eingetragen. Die Anfangsadresse des alten Bausteins im DB 0 bleibt erhalten, d.h. der ursprüngliche Datenbaustein ist weiterhin gültig.
Die Eintragung der Anfangsadresse in den DB 0 erfolgt erst dann, wenn die Übertragung vollständig abgeschlossen ist und alle Kennungen im Bausteinkopf richtig eingetragen sind. Der duplizierte Baustein wird also vom Systemprogramm erst nach der kompletten Übertragung als gültig bzw. vorhanden erkannt.

Hinweis
Das **Verschieben** des **DB 0** ins DB-RAM ist nicht möglich, da dieser sich bereits im DB RAM befindet. Sie können den DB 0 jedoch **duplizieren**.

Parameter

1. AKKU-1-L-L

Nummer des zu kopierenden Bausteins,
zulässige Werte: 0 bis 255
(0 nur für DX oder Duplizieren eines DBs)

2. AKKU-1-H-L

Mit dem Wert in AKKU-1-H-L legen Sie fest, ob Sie einen Baustein **verschieben** oder **duplizieren wollen**:

AKKU-1-H-L = 0:

Der Datenbaustein DB (Aufruf OB 255) oder DX (Aufruf OB 254) mit der in AKKU-1-L-L angegebenen Nummer wird in das DB-RAM **verschoben**.

AKKU-1-H-L = Nr. für den neuen Baustein,
zulässige Werte: 1 bis 255

Der Datenbaustein DB (Aufruf OB 255) oder DX (Aufruf OB 254) mit der in AKKU-1-L-L angegebenen Nummer wird in das DB-RAM **dupliziert** und unter der in AKKU-1-H-L hinterlegten Numer in den DB 0 eingetragen.

Die Werte von AKKU-1-L-H und AKKU-1-H-H werden von den OB 254 und OB 255 nicht berücksichtigt und sind daher für die Parametrierung der OBs ohne Bedeutung.

Fehlerfälle

- Der zu verschiebende Datenbaustein ist nicht vorhanden (OB19).
- Der Baustein ist bereits im DB-RAM vorhanden (OB 31) (deshalb die Funktion nur einmal - vorzugsweise im Anlauf - ausführen).
- Der Speicherplatz im DB-RAM ist nicht ausreichend (OB 31).

Im Fehlerfall wird die Funktion nicht ausgeführt. Das Systemprogramm erkennt einen Laufzeitfehler und ruft den **OB 19 oder OB 31** auf. Die weitere Fehlerreaktion hängt von der Programmierung des OB 19 bzw. 31 ab (siehe Abschnitt 5.6.2). Ist der OB 19 bzw. 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen ist im AKKU-1-L eine Fehlerkennung hinterlegt, die den aufgetretenen Fehler näher erläutert.

Beispiel

Es wird angenommen, daß im Anwenderspeicher die Datenbausteine DB 3 und DB 4 definiert sind. Im DB-RAM soll außer DB 0 noch kein Datenbaustein DB vorhanden sein. Die nachfolgende Tabelle zeigt Ihnen, welche Speicherbelegung nach mehrmaligem Aufruf von OB 255 mit den in der Tabelle aufgeführten Parametern ergibt.

Aufruf-Reihen-folge	**Funktion**	**A K K U - 1 -**				**DB im Speicher nach Aufruf**	
		-H-H	**-H-L**	**-L-H**	**-L-L**	**Anwend.-Sp.**	**DB-RAM**
1	Verschieben	ohne Be-deu-tung	0	ohne Be-deu-tung	3	DB 4	DB 3
2	Duplizieren		5		4	DB 4	DB 3, 5
3	Duplizieren		6		5	DB 4	DB 3, 5, 6
4	Verschieben		0		4	kein DB	DB 3, 4, 5, 6

Erweiterter Datenbaustein DX 0

7

Inhalt von Kapitel 7

Erweiterter Datenbaustein DX 0

7

Im nachfolgenden Kapitel erfahren Sie, wofür Sie den Datenbaustein DX 0 einsetzen können und wie er aufgebaut ist. Sie werden informiert, welche Bedeutung die verschiedenen DX-0-Parameter haben, und Sie lernen an Hand von Beispielen, wie Sie einen Datenbaustein DX 0 erstellen bzw. über eine Maske parametrieren können

7.1 Anwendung

Als Anwender können Sie bestimmte Leistungen des Systemprogramms Ihren Erfordernissen anpassen, wenn Sie im DX 0 alternativ zu den Standard-Voreinstellungen (in der Parameter-Tabelle mit "V" gekennzeichnet) durch Parametereingabe andere Einstellungen vornehmen.

Die Voreinstellungen des Systemprogramms (V) werden automatisch bei jedem NEUSTART gesetzt. Danach wird der DX 0 ausgewertet. Haben Sie den Baustein DX 0 nicht geladen, so gelten weiterhin die Voreinstellungen; ansonsten werden die Einstellungen gültig, mit denen Sie den DX 0 parametriert haben.

Sie können die Einstellungen im DX 0 vornehmen, indem Sie die Werte wie bei anderen Datenbausteinen über STEP-5-Anweisungen programmieren (siehe Abschnitte 7.2 bis 7.4.1) oder – mit PG-Systemsoftware S5-DOS ab Version 3.0 – die Werte über eine spezielle Maske an Ihrem PG als Parameter eingeben (siehe Abschnitt 7.4.2).

Hinweis
Eine Eingabe oder eine Änderung des DX 0 wird nur über einen NEUSTART wirksam.
Wird bei NEUSTART ein **geänderter DX 0** übernommen, so bleiben die **nicht geänderten** Parametrierungen **erhalten!**

7.2 Aufbau des DX 0

Der DX 0 setzt sich aus drei Teilen zusammen:

- der Anfangskennung für den DX 0 (DW 0, 1 und 2)
- mehreren Blöcken unterschiedlicher Länge (je nach Parameteranzahl)

  und

- der Endekennung EEEE.

Anfangskennung

ASCII-Zeichen MASKX0 in DW 0 bis DW 2

Block

Ein **Block** im DX 0 besteht aus 1 bis n Datenwörtern. Diese enthalten:

- die Blockkennung,
- die Blocklänge

  und

- die Blockparameter.

Die **Blockkennung** gibt an, welche Bedeutung die folgenden Parameter haben. Jeder Block ist einem bestimmten Systemprogrammteil oder einer bestimmten Systemfunktion zugeordnet (z.B. leitet die Blockkennung ’04’ den Parameterblock für zyklische Programmbearbeitung ein).

Blocklänge

Die **Blocklänge** gibt an, wieviele Datenwörter die nachfolgenden Parameter belegen.

Parameter

Die möglichen **Parameter** finden Sie in Kapitel 7.3.

Die angegebenen Zahlenwerte sind im Hexadezimalformat (KH).

Endekennung

Sie kennzeichnet das Ende des DX 0 durch EEEEH im letzten Datenwort.

Formaler Aufbau:

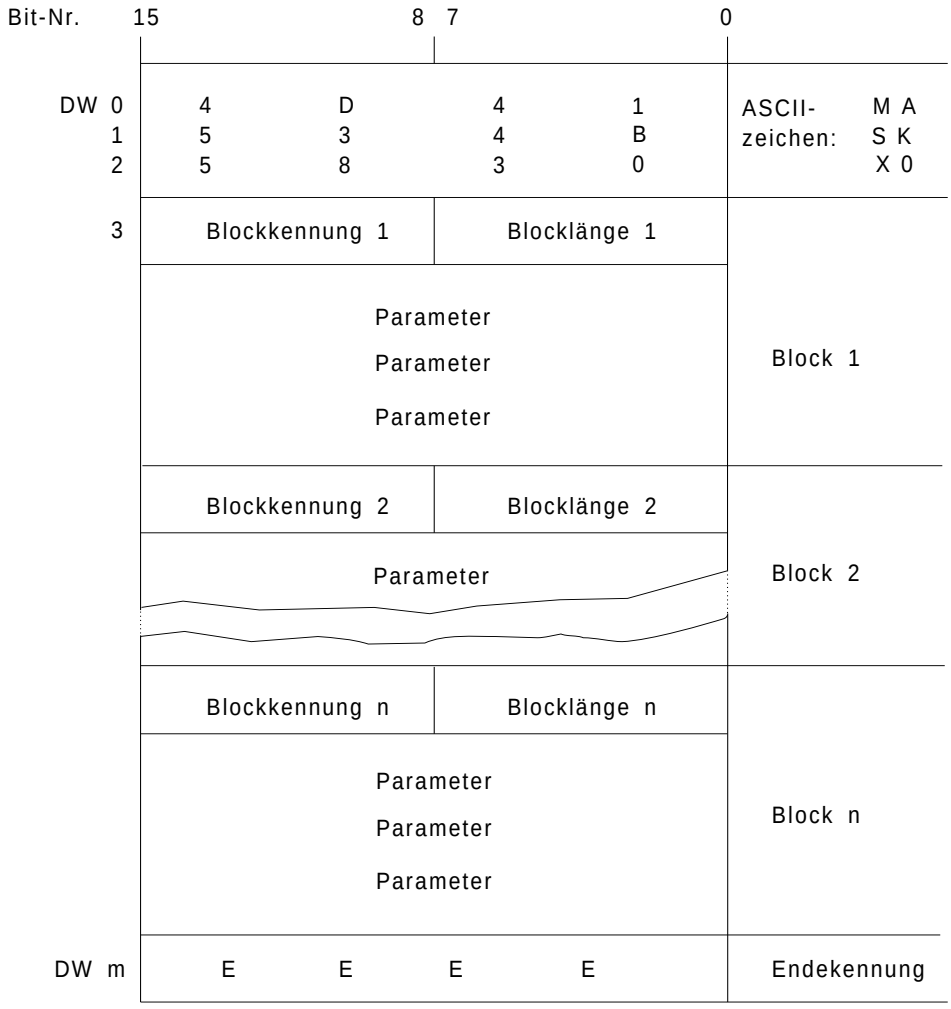


Bild 7-1 Aufbau des DX 0

7.2.1 Beispiel für Eingabe des DX 0

Anfangskennung	DW 0:	KH = 4D41	
	DW 1:	KH = 534B	
	DW 2:	KH = 5830	
Blockkennung/-länge	DW 3:	KH = 0101	Block 1
Parameter (belegt 1 DW)	DW 4:	KH =1001	
Blockkennung/-länge	DW 5:	KH = 0402	
Parameter (belegt 2 DW)	DW 6:	KH = 1000	Block 2
	DW 7:	KH = 0400	
Endekennung	DW10:	KH = EEEE	

Beachten Sie bei der Parametrierung des DX 0 folgende Punkte:

- Die Reihenfolge, in der Sie die einzelnen Blöcke eingeben, ist beliebig.
- Nicht benötigte Blöcke brauchen nicht angegeben zu werden.
- Ist ein bestimmter Block mehrfach vorhanden, gilt der Block, den Sie zuletzt eingegeben haben.
- Die Reihenfolge, in der Sie die einzelnen Parameter eingeben, ist beliebig.
- Nicht benötigte Parameter brauchen nicht angegeben zu werden.
- Ist ein bestimmter Parameter mehrfach genannt, gilt die letzte Angabe.

7.3 Parameter für DX 0

Tabelle 7-1 DX-0-Parameter und ihre Bedeutung

Block-kennung/-länge	Parameter 1. Wort/2. Wort	Bedeutung [1]
ANLAUF und RUN		
02xx [2]	1000	V AUTOMATISCHER WIEDERANLAUF nach NETZ EIN
	1001	AUTOMATISCHER NEUSTART nach NETZ EIN
	2000	V Synchronisation ANLAUF im Mehrprozessorbetrieb
	2001	Keine Synchronisation ANLAUF im Mehrprozessor-betrieb
	3000	V Adressierfehlerüberwachung
	3001	Keine Adressierfehlerüberwachung
	4000	V WIEDERANLAUF
	4001	NEUSTART MIT GEDÄCHTNIS
	6000	V Gleitpunktarithmetik mit 16-bit-Mantisse (geschwin-digkeitsoptimiert)
	6001	Gleitpunktarithmetik mit 24-bit-Mantisse (genauigkeits-optimiert)
	BB00 yyyy	Anzahl der zu aktualisierenden Zeitzellen [3] Voreinstellung: yyyy = 256 Zeitzellen, d.h. Zeitzelle 0 bis 255 zulässig: 0...256
Zyklische Programmbearbeitung		
04xx	1000 yyyy	Länge der Zyklusüberwachungszeit in Millisekunden; Voreinstellung: yyyy = 150 ms, zulässig: 1≤ yyyy ≤ 32C8 (hex) 1 ms bis 13000 ms (dez)
	4000	V Aktualisierung des Prozeßabbildes und der Koppelmerker ohne Semaphorschutz
	4001	Aktualisierung des Prozeßabbildes und der Koppelmerker mit Semaphorschutz (im Block, siehe Abschnitt 10.1.3)

Block-kennung/-länge	Parameter 1. Wort/2. Wort	Bedeutung [1]
Fortsetzung 1 der Tabelle 7-1:		
Alarmbearbeitung: Weckalarme		
06xx	[4]	Auswahl des Bearbeitungs-Modus [4]
	2000	V Prozeßalarm-Signal, pegelgetriggert
	2001	Prozeßalarm-Signal, flankengetriggert
Fehlerbehandlung		
10xx		Weckfehlerbehandlung:
	1000	V Systemstopp, wenn Ereignis eintritt und der OB 33 nicht geladen ist.
	1001	Kein Systemstopp, wenn Ereignis eintritt und der OB 33 nicht geladen ist.
		Reglerfehlerbehandlung:
	1200	V Systemstopp, wenn Ereignis eintritt und der OB 34 nicht geladen ist.
	1201	Kein Systemstopp, wenn Ereignis eintritt und der OB 34 nicht geladen ist.
		Zyklusfehlerbehandlung:
	1400	V Systemstopp, wenn Ereignis eintritt und der OB 26 nicht geladen ist.
	1401	Kein Systemstopp, wenn Ereignis eintritt und der OB 26 nicht geladen ist.
		Befehlscodefehlerbehandlung:
	1800	V Systemstopp, wenn Ereignis eintritt und der OB 27/29/30 nicht geladen ist.
	1801	Kein Systemstopp, wenn Ereignis eintritt und der OB 27/29/30 nicht geladen ist.
		Laufzeitfehlerbehandlung:
	1A00	V Systemstopp, wenn Ereignis eintritt und der OB 19/31/32 nicht geladen ist.
	1A01	Kein Systemstopp, wenn Ereignis eintritt und der OB 19/31/32 nicht geladen ist.

Block-kennung/-länge	Parameter 1. Wort/2. Wort	Bedeutung [1]
Fortsetzung 2 der Tabelle 7-1:		
10xx		Adressierfehlerbehandlung:
	1C00	V Systemstopp, wenn Ereignis eintritt und der OB 25 nicht geladen ist.
	1C01	Kein Systemstopp, wenn Ereignis eintritt und der OB 25 nicht geladen ist.
		Quittungsverzugsfehlerbehandlung:
	1E00	Systemstopp, wenn Ereignis eintritt und der OB 23/24 nicht geladen ist.
	1E01	V Kein Systemstopp, wenn Ereignis eintritt und der OB 23/24 nicht geladen ist.
		Schnittstellenfehlerbehandlung:
	2000	Systemstopp, wenn Ereignis eintritt und der OB 35 nicht geladen ist.
	2001	V Kein Systemstopp, wenn Ereignis eintritt und der OB 35 nicht geladen ist.
EEEE		Endekennung

[1] V = Voreinstellung bei nicht geladenem DX 0 bzw. fehlendem Block

[2] xx = Blocklänge (Anzahl der von den Parametern belegten Datenwörter)

[3] Zur Aktualisierung der Zeitzellen lesen Sie bitte die Erklärung auf der nächsten Seite

[4] Parameter und Bedeutung entnehmen Sie bitte der auf Seite 7 - 9 aufgeführten Tabelle

Hinweis
Die aktuelle PG-Software (STEP 5/ST Vers. 6 bzw. STEP 5/MT Vers. 2) zum Generieren des DX 0 mit Hilfe einer Maske setzt die Parameter für die **Schnittstellenfehlerbehandlung** (2000 bzw. 2001) und für die Auswahl **"Wiederanlauf oder Neustart mit Gedächtnis"**(4000 bzw. 4001) nicht ab.
Sie können diese Parameter z. B. mit der PG-Funktion "Ausgabe Baustein" eintragen (vergessen Sie dabei nicht, die Blocklänge zu ändern!). Ein so geänderter DX 0 ist mit der aktuellen PG-Software mit Hilfe der Funktion "Ausgabe Maske nicht mehr editierbar.

Aktualisierung der Zeitzellen

- Standardmäßig werden die Zeitzellen T 0 bis T 255 aktualisiert.
- Wenn Sie im DX 0 die Anzahl ’0’ eintragen, werden **keine** Zeitzellen aktualisiert, auch dann nicht, wenn Sie im Programm enthalten sind. Es gibt dann auch keine Fehlermeldung.

Es wird immer eine **gerade** Anzahl an Zeitzellen aktualisiert:

Eintrag	’0’	’1’und ’2’	’3’ und ’4’	’5’ und ’6’	’7’ und ’8’	....
Aktualisierung	keine	T0 und T1	T0 bis T3	T0 bis T5	T0 bis T7	....

Hinweis
Die Anzahl der Zeitzellen können Sie auch im Datenbaustein DB 1 parametrieren (siehe Abschnitt 10.1.6). Wir empfehlen jedoch, diesen Paramter **nur im DX 0** anzugeben.
Falls Sie so**wohl im DX 0 als auch im DB 1** die Anzahl der Zeitzellen einstellen, gilt der Wert, den Sie im **DB 1** angegeben haben!

Parameter für die Alarmbearbeitung

Aus der folgenden Tabelle können Sie für Ihre Alarmbearbeitung den passenden Parameter heraussuchen und den DX 0 damit programmieren. Je nachdem, welchen Parameter Sie wählen, sind entsprechend den Symbolen bestimmte (oder alle) Alarme an Bausteingrenzen, andere (oder alle) an Befehlsgrenzen wirksam:

Parameter/ (alt) [1]	Zeitalarm	Weckalarme									Regleralarm	Verz.-Alarm	Proz.-Alarm
		5 s	2 s	1 s	500 ms	200 ms	100 ms	50 ms	20 ms	10 ms			
122C V (100C)	□	□	□	□	□	□	□	□	□	□	□	□	□
1224 (100A)	□	□	□	□	□	□	□	□	□	□	□	□	■
1220	□	□	□	□	□	□	□	□	□	□	□	■	■
121C (1008)	□	□	□	□	□	□	□	□	□	□	■	■	■
1216	□	□	□	□	□	□	□	□	□	■	■	■	■
1214	□	□	□	□	□	□	□	□	■	■	■	■	■
1212	□	□	□	□	□	□	□	■	■	■	■	■	■
1210	□	□	□	□	□	□	■	■	■	■	■	■	■
120E	□	□	□	□	□	■	■	■	■	■	■	■	■
120C	□	□	□	□	■	■	■	■	■	■	■	■	■
120A	□	□	□	■	■	■	■	■	■	■	■	■	■
1208	□	□	■	■	■	■	■	■	■	■	■	■	■
1206	□	■	■	■	■	■	■	■	■	■	■	■	■
1204 (1006)	■	■	■	■	■	■	■	■	■	■	■	■	■

V = Voreinstellung

□ Unterbrechung an Bausteingrenzen

■ Unterbrechung an Befehlsgrenzen

Hinweis

Falls die Alarmbearbeitung an Befehlsgrenzen eingeschaltet ist, werden bei Unterbrechungen u. U. auch die Operationen ‘TNB’ bzw. ‘TNW’ unterbrochen. Dies gilt ebenso für einige wenige Sonderfunktions-Organisationsbausteine, Standardfunktionsbausteine und Regler-Funktionsbausteine.

[1] Die PG-Software zum Generieren des DX 0 setzt die "alten" Parameter ein. Soll ein DX 0 , der über STEP 5 mit neuen Parametern erstellt wurde, am PG angezeigt werden, so führt dies zu einer Fehlermeldung.

7.4 Parametrierungsbeispiele

7.4.1 STEP-5-Programmierung

Beispiel A:

Sie wollen im Mehrprozessorbetrieb drei CPUs einsetzen: CPU A, B und C. CPU A und B arbeiten eng miteinander zusammen, tauschen häufig Daten aus und bearbeiten ein umfangreiches Anlaufprogramm. CPU C bearbeitet weitgehend unabhängig davon ein kurzes, zeitkritisches Programm.

Standardmäßig beginnen im Mehrprozessorbetrieb alle CPUs gemeinsam mit der zyklischen Programmbearbeitung, d.h., die CPUs warten solange aufeinander, bis alle ihren Anlauf beendet haben, und gehen dann gemeinsam in die zyklische Programmbearbeitung.

Da CPU C ihr Programm unabhängig von den anderen CPUs ausführt und ein **sehr kurzes Anlaufprogramm** bearbeitet, ist bei ihr keine Synchronisation des Anlaufs notwendig. Durch Parametrierung des DX 0 erreichen Sie, daß CPU C nach beendetem Anlauf sofort in die zyklische Programmbearbeitung geht, ohne auf CPU A und B zu warten.

Programmieren Sie den DX 0 für CPU C:

DX 0	Anfangskennung "MASKX0"	DW 0:	KH = 4D41
		DW 1:	KH = 534B
		DW 2:	KH = 5830
	1. Blockkennung/-länge	DW 3:	KH = 0201
	Parameter 1	DW 4:	KH = 2001
	Endekennung	DW 5:	KH = EEEE

Haben Sie diesen DX 0 in den Programmspeicher geladen, wird er mit dem nächsten NEUSTART wirksam. Da CPU C ein sehr kurzes Anlaufprogramm bearbeitet und nicht auf A und B wartet, geht bei ihr sofort nach dem Anlauf die grüne RUN-LED an. Das BASP-Signal (Befehlsausgabesperre) wird jedoch erst inaktiv geschaltet, wenn alle drei CPUs ihren Anlauf beendet haben. Dies bedeutet, daß CPU C nicht auf die Digitalperipherie zugreifen darf.

Beispiel B:

Mit der folgenden Parametrierung des DX 0 wird

- die Adressierfehler-Überwachung abgeschaltet,
- die Zeitzellenaktualisierung abgeschaltet,
- die Zykluszeit auf 4 s eingestellt.

DX 0	Anfangskennung "MASKX0"	DW 0:	KH = 4D41
		DW 1:	KH = 534B
		DW 2:	KH = 5830
	1. Blockkennung/-länge	DW 3:	KH = 0203
	Parameter	DW 4:	KH = 3001
	Parameter [1)]	DW 5:	KH = BB00
		DW 6:	KH = 0000
	2. Blockkennung/-länge	DW 7:	KH = 0402
	Parameter [1)]	DW 8:	KH = 1000
		DW 9:	KF = +4000
	Endekennung	DW10:	KH = EEEE

Diese Parametrierung des DX 0 hat folgende Auswirkungen auf die Programmbearbeitung:

- Derjenige Teil des Prozeßabbilds, dem keine Peripheriebaugruppen zugeordnet sind, kann als zusätzlicher "Merkerbereich" benutzt werden.
- Die Laufzeit des Systemprogramms verkürzt sich, da keine Zeitzellen aktualisiert werden.
- Ein Zyklusfehler wird erst dann erkannt, wenn die Laufzeit des Anwenderprogramms und des Systemprogramms zusammen 4 s übersteigt.

[1)] Parameter, die zwei Datenwörter belegen, müssen bei der Angabe der Blocklänge mit ’2’ berücksichtigt werden!

7.4.2 Parametrierung über PG-Maske

Für die Parametrierung des DX 0 stehen Ihnen bei der PG-Systemsoftware S5-DOS (ab Version 3) Masken zur Verfügung. Die PG-Software generiert entsprechend den Parametervorbesetzungen und den von Ihnen eingegebenen Parametern automatisch den Datenbaustein DX 0. Für die Parametrierung werden zwei Masken benötigt.

Die grundsätzlichen Bedienungen zur Anwahl und zum Ausfüllen von PG-Masken entnehmen Sie bitte Ihrer PG-Beschreibung.

Hantierung beim Ausfüllen der DX-0-Masken:

Die PG-Maske zum Ausfüllen des DX 0 besteht aus zwei Teilmasken:

Die erste Teilmaske (BIld 7-2) enthält die Parametergruppe

ANLAUF NACH NETZ EIN,
MEHRPROZESORANLAUF SYNCHRONOSIEREN,
BLOCKUEBERTRAGUNG DER KOPPELMERKER,
ADRESSIERFEHLERUEBERWACHUNG,
ZYKLUSZEITUEBERWACHUNG,
ANZAHL DER ZEITZELLEN,
GENAUIGKEIT DER GLEITPUNKTARITHMETIK.

DX0-PARAMETRIERUNG (AG 135U: CPU 928, R-PROZESSOR) DX 0

ANLAUFART NACH "NETZ EIN":	***1***	(1 = WIEDERANLAUF 2 = NEUSTART)
MEHRPROZESSORANLAUF SYNCHRONISIEREN	***JA***	
BLOCKUEBERTRAGUNG DER KOPPELMERKER	***NEIN***	
ADRESSIERFEHLERUEBERWACHUNG	***JA***	
ZYKLUSZEITUEBERWACHUNG (X 10 MS):	***15***	(R-PROZ: 1 - 400 CPU 928: 1 - 600)
ANZAHL DER ZEITZELLEN	***256***	(R-PROZ: 0 - 128 CPU 928: 0 - 256)
GENAUIGKEIT DER GLEITPUNKTARITHMETIK #24-BIT-MANTISSE NUR BEI CPU 928#	***16 - BIT - MANTISSE***	

F1	F2	F3	F4	F5	F6	F7	F8
		WAEHLEN			WEITER		

Bild 7-2 PG-Maske zum Parametrieren des DX 0 / Teil 1

Haben Sie die notwendigen Parametereinstellungen in der 1. Teilmaske durchgeführt oder finden Sie die zu ändernden Parameter dort nicht, so können Sie sich die zweite Teilmaske (Bild 7-3) mit folgender Parametergruppe anzeigen lassen:

ADRESSIERFEHLER, ZYKLUSFEHLER,
QUITTUNGSFEHLER, WECKFEHLER
BEFEHLSCODEFEHLER, REGLERFEHLER,
LAUFZEITFEHLER,
PROZESSALARMBEARBEITUNG,
UNTERBRECHBARKEIT DES ANWENDERPROGRAMMS
DURCH ALARME.

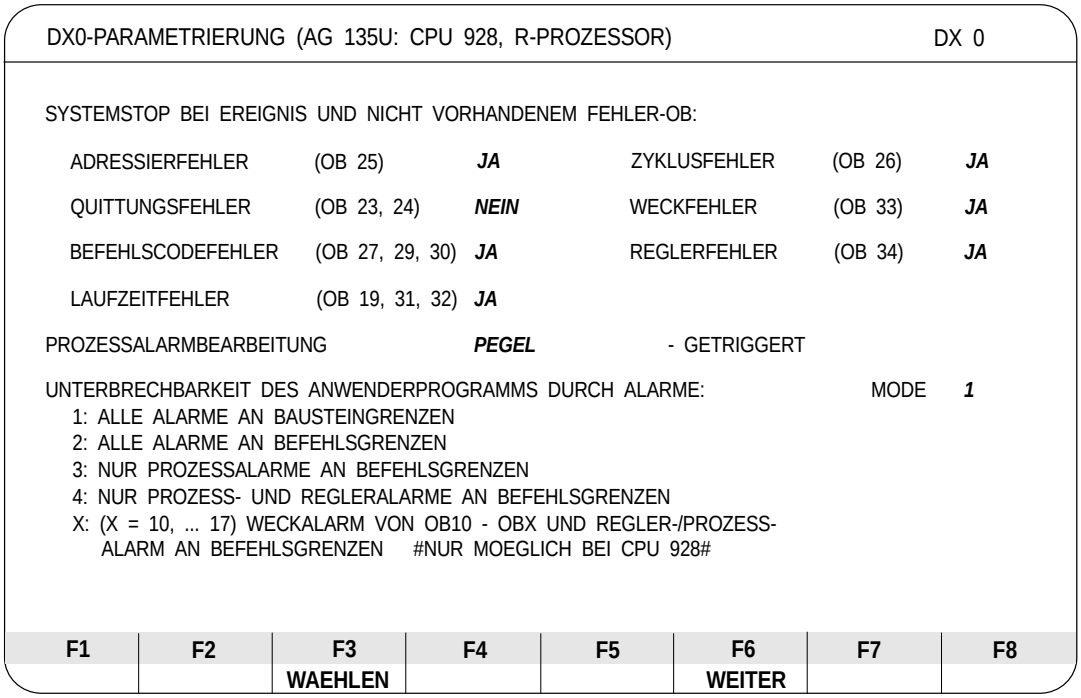
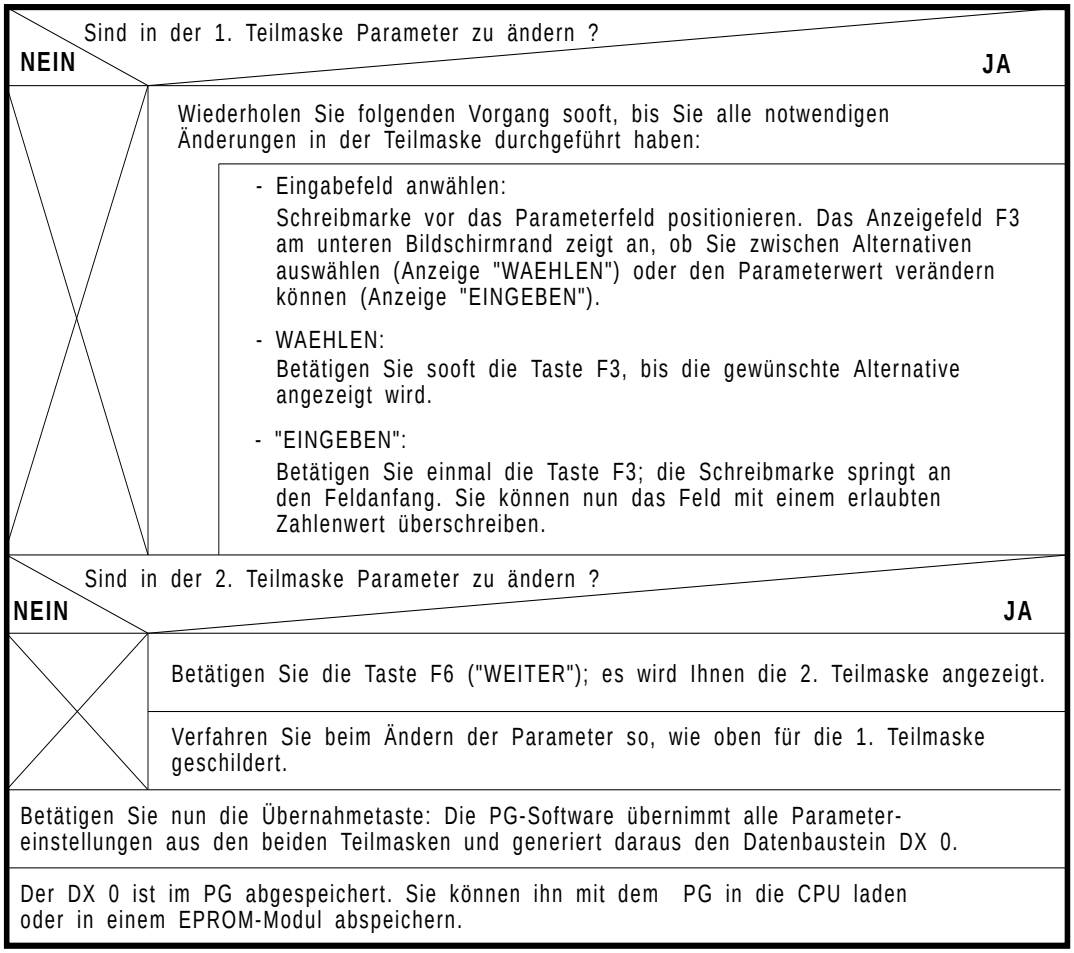


Bild 7-3 PG-Maske zum Parametrieren des DX 0 / Teil 2

Das nachfolgende Ablaufdiagramm erläutert Ihnen, wie Sie die Teilmasken und Maskenfelder ausfüllen, die Parameter abspeichern und den erzeugten Datenbaustein DX 0 laden können.

Ablaufdiagramm zum
Ausfüllen der DX-0-Masken:



Ein Beispiel zum Ausfüllen finden Sie auf der folgenden Seite.

Beispiel zum Ausfüllen der DX-0-Maske:

Sie wollen den DX 0 für folgendes Verhalten des Systemprogramms abweichend von den Vorbesetzungen parametrieren:

- Im Mehrprozessorbetrieb soll die CPU, für die der DX 0 programmiert wird, im Anlauf nicht warten bis die übrigen CPUs ihren Anlauf beendet haben.
- Die Zyklusüberwachungszeit soll 100 ms betragen.
- Arithmetische Operationen sollen mit 24-Bit-Gleitpunktmantisse durchgeführt werden.
- Bei Zyklusfehlern soll die CPU **nicht** in STOP gehen, wenn der OB 26 nicht geladen ist.
- Das Anwenderprogramm soll von allen Alarmen an Befehlsgrenzen unterbrochen werden.

Füllen Sie für diese Wünsche die Maske folgendermaßen aus:

Erste DX-0-Maske:

- Wählen Sie bei Parameter MEHRPROZESSORANLAUF SYNCHRONISIEREN mit der Funktionstaste F3 NEIN an.
- Betätigen Sie beim Parameter ZYKLUSZEITUEBERWACHUNG zunächst die Funktionstaste F3 und geben Sie die Ziffer 10 (= 100 ms) ein.
- Wählen Sie beim Parameter GENAUIGKEIT DER GLEITPUNKTARITHMETIK mit der Funktionstaste F3 die "24-Bit-Mantisse" an.
- Betätigen Sie die Funktionstaste F6 (WEITER). Es wird danach die zweite DX-0-Maske vorgelegt

Zweite DX-0-Maske:

- Wählen Sie bei Parameter ZYKLUSFEHLER mit der Funktionstaste F3 NEIN an.
- Geben Sie in das Feld MODE bei Parameter UNTERBRECHBARKEIT DES ANWENDERPROGRAMMS DURCH ALARME die Ziffer ’2’ (= alle Alarme an Befehlsgrenzen ein)
- Bestätigen Sie mit der Übernahmetaste die gesamten Eingaben. Der Datenbaustein DX 0 wird nun von der PG-Software generiert.

Übertragen Sie anschließend den DX 0 in den Speicher der CPU oder in ein EPROM-Speichermodul.

Speicherbelegung und Speicherorganisation

8

Inhalt von Kapitel 8

8 Speicherbelegung und Speicherorganisation

In diesem Kapitel können Sie nachschlagen, wie der Speicher der CPU 928B organisiert ist. Sie finden ferner wichtige und für den Anwender zugängliche Informationen, die in einigen Systemdatenwörtern hinterlegt sind.

8.1 Struktur des Speicherbereiches

Der Speicherbereich in der CPU 928B enthält im wesentlichen folgende Bereiche:

Tabelle 8-1 Struktur des Speicherbereiches

Speicherbereich		Länge	Datenbreite
Anwenderspeicher für:	OB, FB, FX, PB, SB, DB, DX	max. $32 * 2^{10}$ Wörter	16 bit
DB-RAM für:	Datenbausteine, Schieberegister	$23 * 2^{10}$ Wörter	16 bit
Merker:	S	1024 byte	8 bit
Bereich Anschaltung: Bereich System: Zähler: Zeiten:	BA, BB BS, BT Z T	je 256 Wörter je 256 Wörter 256 Wörter 256 Wörter	16 bit 16 bit 16 bit 16 bit
Merker:	M	256 byte	8 bit
Prozeßabbild (PA) der Ein- und Ausgänge:	PAE, PAA	je 128 byte	8 bit
Peripheriebereich, unterteilt in:			8 bit
	P-Peripherie	256 byte	
	Q-Peripherie	256 byte	
	IM 3	256 byte	
	IM 4	256 byte	
	Koppelmerker	256 byte	
	Koordinatorbaugruppe (KOR)	256 byte	
	Kacheln (CP, IP, KOR 923C)	2048 byte	
	Dezentrale Peripherie	768 byte	

Die genauen Adressen dieser Bereiche entnehmen Sie den Speicherbelegungsplänen im nächsten Abschnitt.

Hinweis
Der STEP-5-Zugriff auf eine Speicherzelle innerhalb eines Operandenbereiches (z.B. Merker) sollte nie direkt über die absolute Adresse dieser Speicherzelle erfolgen, sondern ausschließlich relativ zur Basisadresse des jeweiligen Operandenbereichs.
Die Basisadressen aller Operandenbereiche sind im Bereich der Systemdaten (BS-Bereich) abgelegt (siehe "Systemdatenbelegung").

8.2 Adreßraumaufteilung der CPU 928B

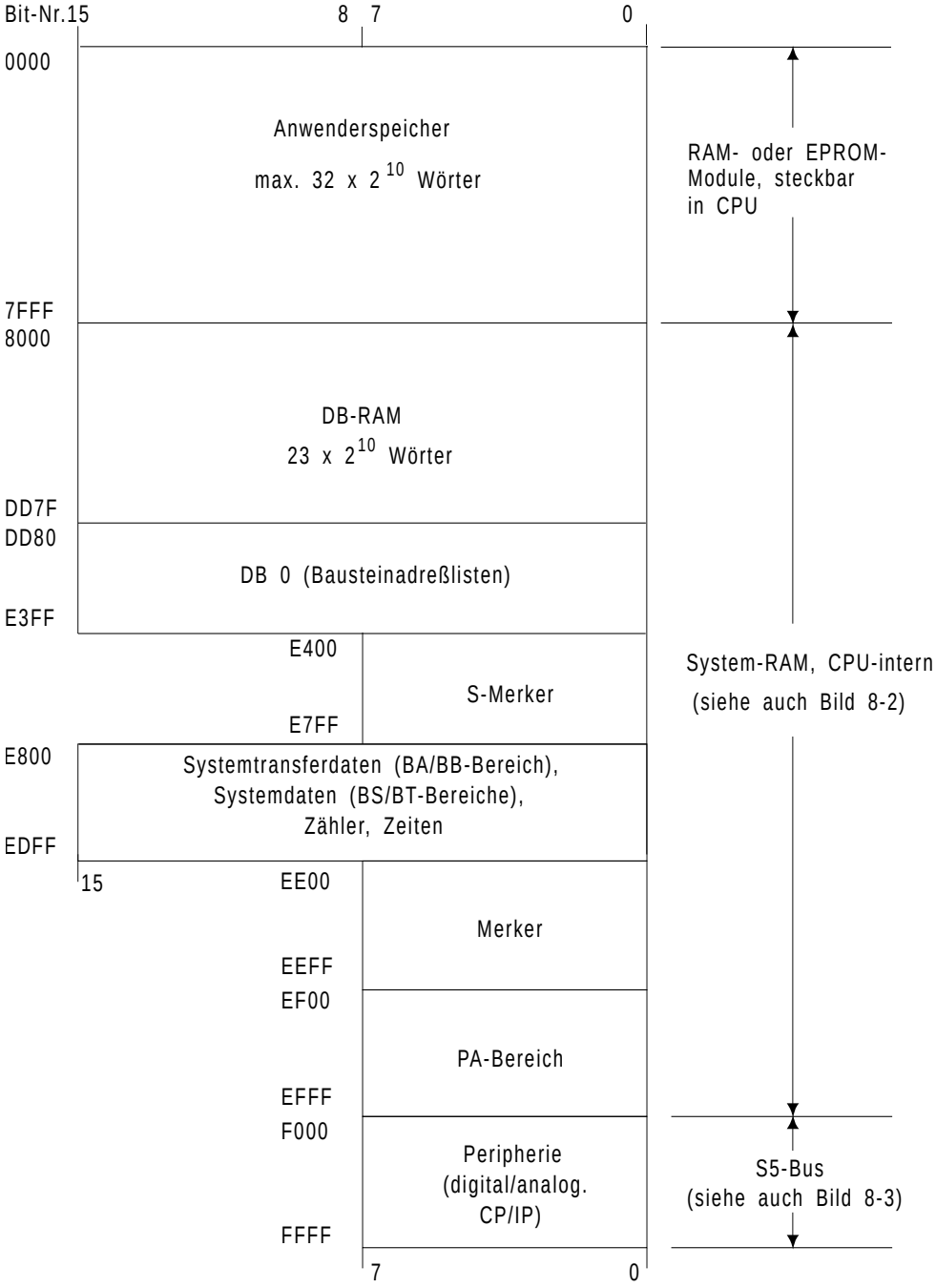


Bild 8-1 Adreßraumaufteilung der CPU 928B/Übersicht

8.2.1 Adreßraumaufteilung des System-RAMs

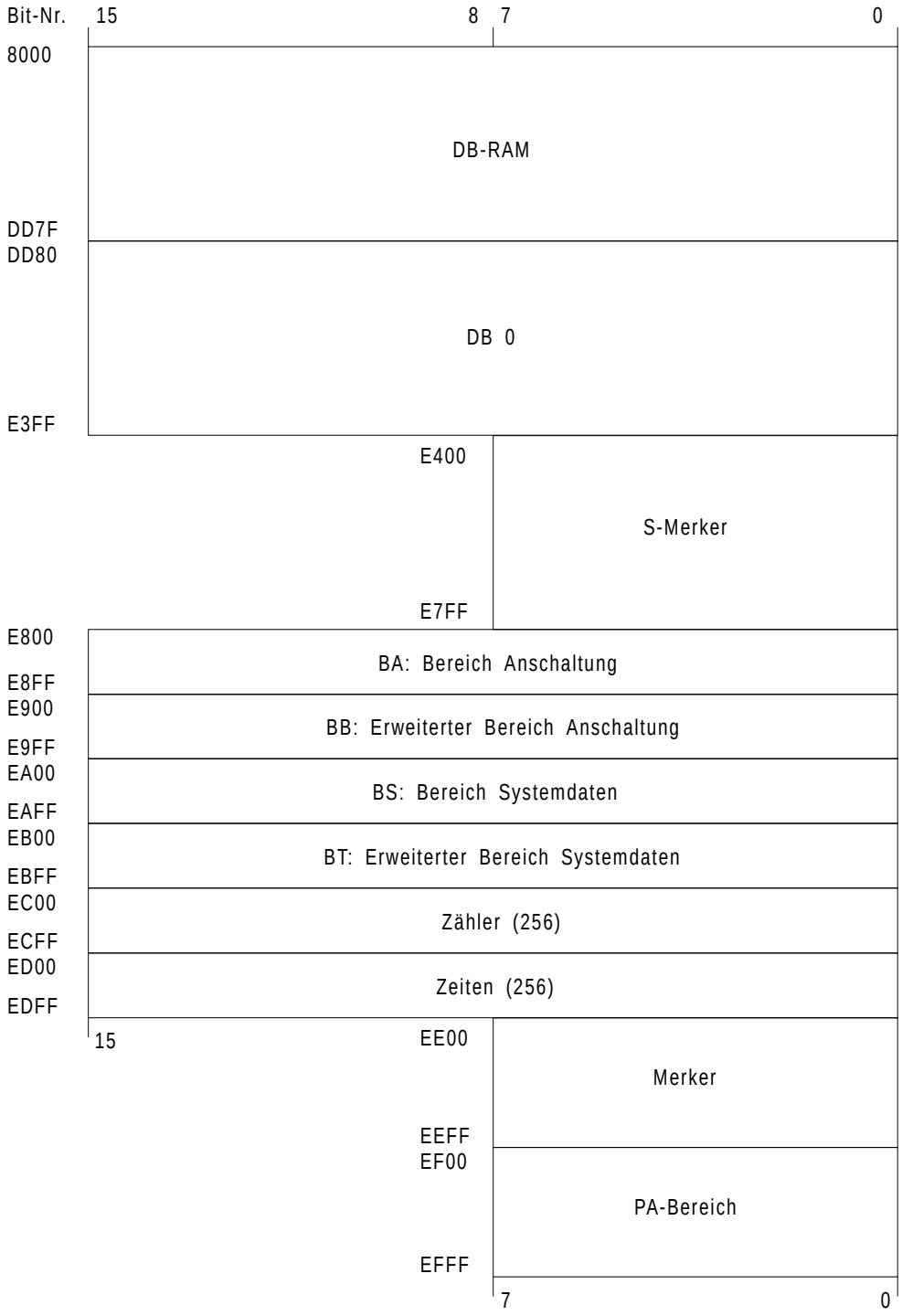


Bild 8-2 Adreßraumaufteilung System-RAM

8.2.2 Adreßraumaufteilung der Peripherie

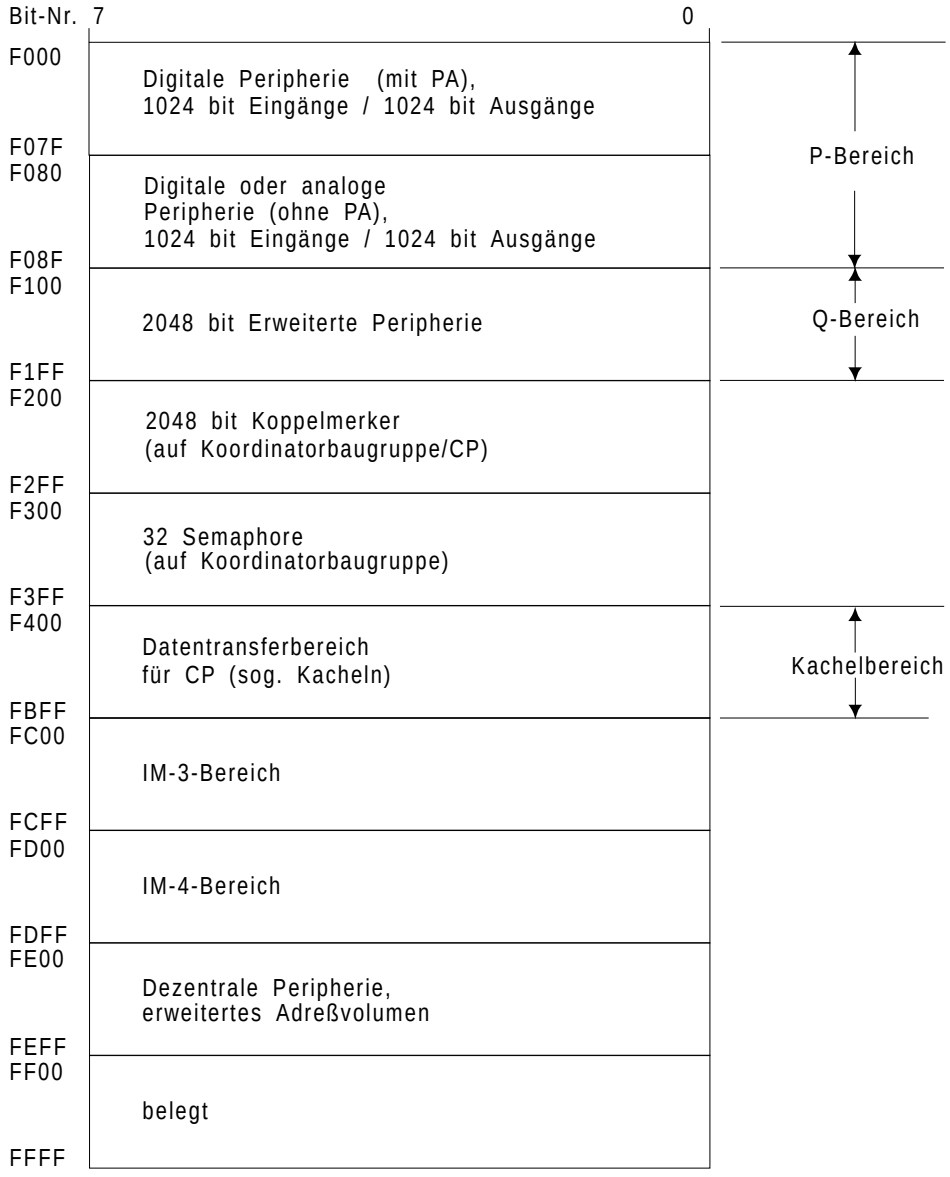


Bild 8-3 Adreßraumaufteilung der Peripherie (8 bit) auf dem S-5-Bus

Adreßbereiche für Peripherie und deren Programmierung

Bereich (absolute Adresse)	Wird angesprochen mit	Parameter
EF00 bis EF7F: PAE (Prozeßabbild Eingänge)	L EB / T EB	0 bis 127
	L EW / T EW	0 bis 126
	L ED / T ED	0 bis 124
	U E / UN E / O E / ON E S E/ R E / = E	0.0 bis 127.7
EF80 bis EFFF: PAA (Prozeßabbild Ausgänge)	L AB / T AB	0 bis 127
	L AW / T AW	0 bis 126
	L AD / T AD	0 bis 124
	U A / UN A / O A / ON A S A / R A / = A	0.0 bis 127.7
P-Peripherie mit Prozeßabbild	Zum Zeitpunkt der Operationsbearbeitung wird nur das Prozeßabbild verändert. Erst am Ende des Zyklus wird der neue Zustand des Prozeßabbilds der Ausgänge an die Peripherie ausgegeben.	
F000 bis F07F: Digitale Peripherie Eingänge/ Ausgänge	L PY / T PY	0 bis 127
	L PW / T PW	0 bis 126
F080 bis F0FF: Digitale oder analoge Peripherie Eingänge/Ausgänge	L PY / T PY	128 bis 255
	L PW / T PW	128 bis 254
P-Peripherie	Die Ein- und Ausgänge werden byte- oder wortweise direkt angesprochen.	
F100 bis F1FF: Erweiterte Peripherie Eingänge/Ausgänge	L QB / T QB	0 bis 255
	L QW / T QW	0 bis 254
Q-Peripherie	Die Ein- und Ausgänge werden byte- oder wortweise direkt angesprochen.	

Mit STEP-5-Operationen können Sie entweder direkt oder über das Prozeßabbild auf die Peripherie zugreifen. Beachten Sie dabei, daß ein Prozeßabbild nur für Ein- und Ausgabebytes der P-Peripherie mit Byteadressen von 0 bis 127 existiert!

Hinweis
Über die Anschaltungen IM 304, IM 307 und IM 308 können Sie mit Ihrem Programm auf dezentrale Adreßbereiche zugreifen. Hierdurch gewinnen Sie zwei neue, dem Q-Bereich gleichwertige Adreßbereiche. Ein Zugriff auf diese Bereiche ist aber im Gegensatz zum Q-Bereich nur über absolute Adressierung oder mit dem FB 196 aus dem Softwarepaket "Grundfunktionen" (siehe Katalog ST59) möglich.

8.3 Organisation des Anwenderspeichers in der CPU 928B

Der Anwenderspeicher umfaßt – abhängig vom gesteckten Speichermodul – den Speicherbereich von 0000H bis 7FFFH. Beim Laden der einzelnen Bausteine des Anwenderprogramms werden diese in beliebiger Reihenfolge im Speicher abgelegt (aufsteigende Adressen).

"alternatives Laden" der Datenbausteine

Das Laden der Datenbausteine DB/DX erfolgt abhängig von der Einstellung im Systemdatenwort BS 144 alternativ:
In der Vorbesetzung werden die Datenbausteine zunächst in den Anwenderspeicher geladen. Erst wenn dieser gefüllt ist, werden die Datenbausteine im internen DB-RAM (8000H bis DD7FH) abgelgt. Durch Setzen von Bit 0 in BS 144 können Sie diese Reihenfolge umkehren ("alternatives Laden").

Speicherauskunft

Mit der PG-Online-Funktion SPAUS (Speicherausbau) erhalten Sie die Adresse (hexadezimal) der Speicherzelle, die den Bausteinende-Befehl des letzten im Speichermodul vorhandenen Bausteins enthält, sowie die Größe des RAM-Moduls.

Verwaltung der Bausteine

Beim Korrigieren von Bausteinen wird der "alte" Baustein im Speicher für ungültig erklärt und ein neuer Baustein in den Speicher und in die Adreßliste eingetragen. Ebenso werden beim Löschen von Bausteinen die Bausteine im Speicher nicht wirklich gelöscht, sondern nur für ungültig erklärt. Lücken, die durch Löschen von Bausteinen entstanden sind, werden als freie Speicherplätze verwaltet und beim Laden neuer Bausteine wieder verwendet.

Speicher komprimieren

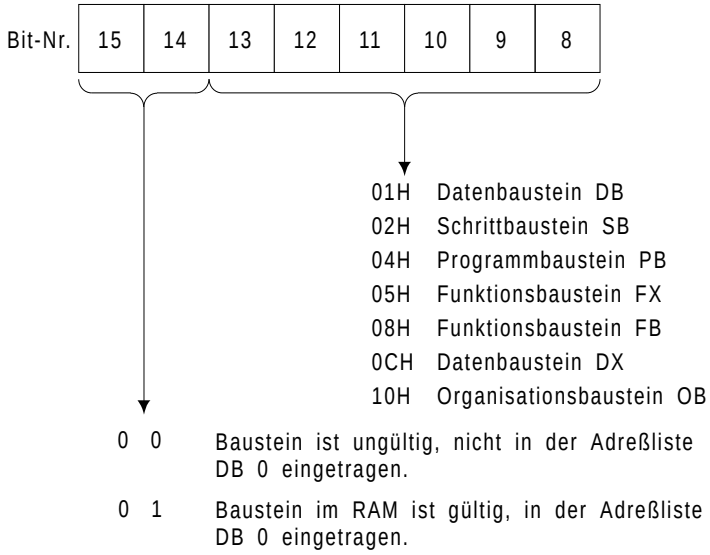
Mit der Online-Funktion SPEICHER KOMPRIMIEREN schaffen Sie Speicherplatz für neue Bausteine: Die Funktion optimiert die Speicherplatzbelegung, indem sie ungültig markierte Bausteine löscht und die gültigen Bausteine zusammenschiebt. Das Zusammenschieben erfolgt getrennt nach Speichermodul und internem RAM-Modul (siehe Abschnitt 11.2.2).

8.3.1 Bausteinköpfe im Anwenderspeicher

Jeder Baustein im Speicher beginnt mit einem 5 Wörter langen Bausteinkopf.

1. Wort: Baustein-Anfangskennung: 7070H

2. Wort: High-Byte = Bausteintyp



Low-Byte = Bauteinnummer
Die Bausteinnumer (0 bis 255) liegt im Low-Byte des 2. Kopfwortes und wird als Dualzahl codiert: 00 bis FFH.

3. Wort: Im High-Byte des 3. Wortes stehen die Kennungen für das Programmiergerät, im Low-Byte ein Teil der Bibliotheksnummer.

4. Wort: Das vierte Wort enthält den Rest der Bibliotheksnummer.

5. Wort: Im 5. Wort (Low- und High-Byte) steht die Länge des Bausteins inklusive Bausteinkopf. Die Angabe erfolgt in Wörtern.

8.3.2 Bausteinadreßlisten im Datenbaustein DB 0

Der Datenbaustein DB 0 enthält die Adreßiste mit den Anfangsadressen aller Anwenderbausteine, die sich im Speichermodul oder im DB-RAM der CPU befinden. Diese Adreßliste wird nach NETZ EIN vom Systemprogramm bei NEUSTART aufgebaut, bei WIEDERANLAUF überprüft und bei jeder Eingabe oder Änderung von Bausteinen mit dem PG automatisch aktualisiert.

Anfangsadressen der Adreßlisten

Für jeden Bausteintyp gibt es im DB 0 eine eigens reservierte, 256 Wörter lange Adreßliste, d. h. für jeden Baustein ist 1 Wort rerserviert. Nicht geladene und gelöschte Bausteine haben die Anfangsadresse ’0’.

Die Anfangsadressen der einzelnen Bausteinadreßlisten stehen außerdem in den Systemdaten BS 32 bis BS 38:

BS 32: Anfangsadresse der DX-Adreßliste

BS 33: Anfangsadresse der FX-Adreßliste

BS 34: Anfangsadresse der DB-Adreßliste

BS 35: Anfangsadresse der SB-Adreßliste

BS 36: Anfangsadresse der PB-Adreßliste

BS 37: Anfangsadresse der FB-Adreßliste

BS 38: Anfangsadresse der OB-Adreßliste (nur 48 Wörter lang)

Baustein-Anfangsadressen

Die Baustein-Anfangsadressen in den Adreßlisten zeigen immer auf das erste Wort **nach** dem Bausteinkopf:

- bei Datenbausteinen jeweils auf das Datenwort DW 0.
- bei Codebausteinen jeweils auf die erste STEP-5-Anweisung (bei FBs auf den ’SPA’-Befehl vor dem Namen und der Parameterliste).

Ablage der Bausteinadressen im DB 0

n = Anfangsadresse der PB-Adreßliste (= Inhalt von BS 36)

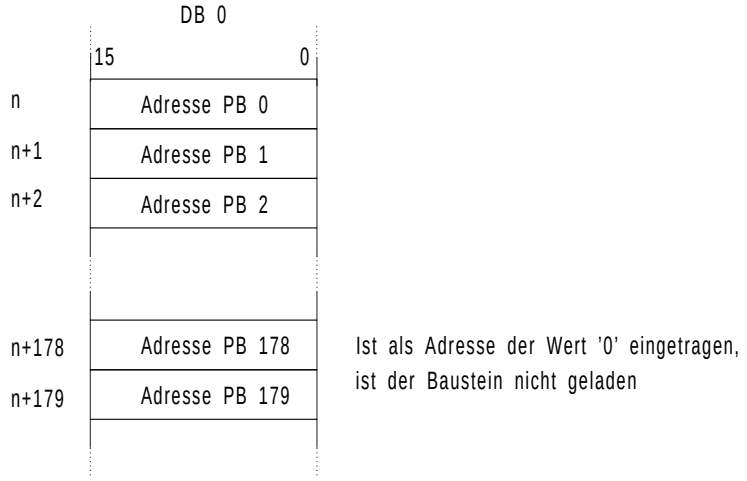


Bild 8-4 Bausteinadressen im DB 0

Beispiele für die Ermittlung einer Bausteinadresse

Anfangsadresse des FB 40

Lösung a):

```
   :L    BS 37   Basisadresse FB-Adressliste
   :L    KB 40   + FB-Nummer
   :+F           = Adresse der Speicherzelle, die
   :             die Anfangsadresse des FB 40
   :             enthaelt
   :LIR  1       Anfangsadresse des FB 40 in
   :             den AKKU 1 laden.
   :             (Baustein ist nicht vorhanden,
   :             wenn Anfangsadesse = 0)
```

Lösung b):

```
   :L    BS 37   Basisadresse FB-Adressliste
   :MAB          BR-Register mit Basisadresse laden
   :LRW  +40     Inhalt von Speicherzelle
   :             "Basisadresse + 40" in den
   :             AKKU 1 laden
```

Anfangsadresse und Länge des Datenbausteins DB 50 ermitteln

a) über indirekten Speicherzugriff:

```
      :L    BS 34      Basisadresse der DB-Adressliste laden
      :L    KB 50      Die Adresse des Eintrags für den DB 50
      :+F              errechnen und die Anfangsadresse
      :LIR  1             in den AKKU 1 laden
      :L    KB 0       Bei nicht vorhandenem Baustein zur Marke
      :!=F                NIVO springen
      :SPB  =NIVO
      :ENT             Anfangsadresse des DB 50 in AKKU 3 und
      :TAK                und AKKU 1 laden
      :L KF -1         Anfangsadresse um Eins vermindern und
      :+F                 und die Bausteinlänge in den AKKU 1
      :LIR  1             laden
      : .
      : .
NIVO  : .......        Reaktion, wenn Baustein nicht vorhanden
```

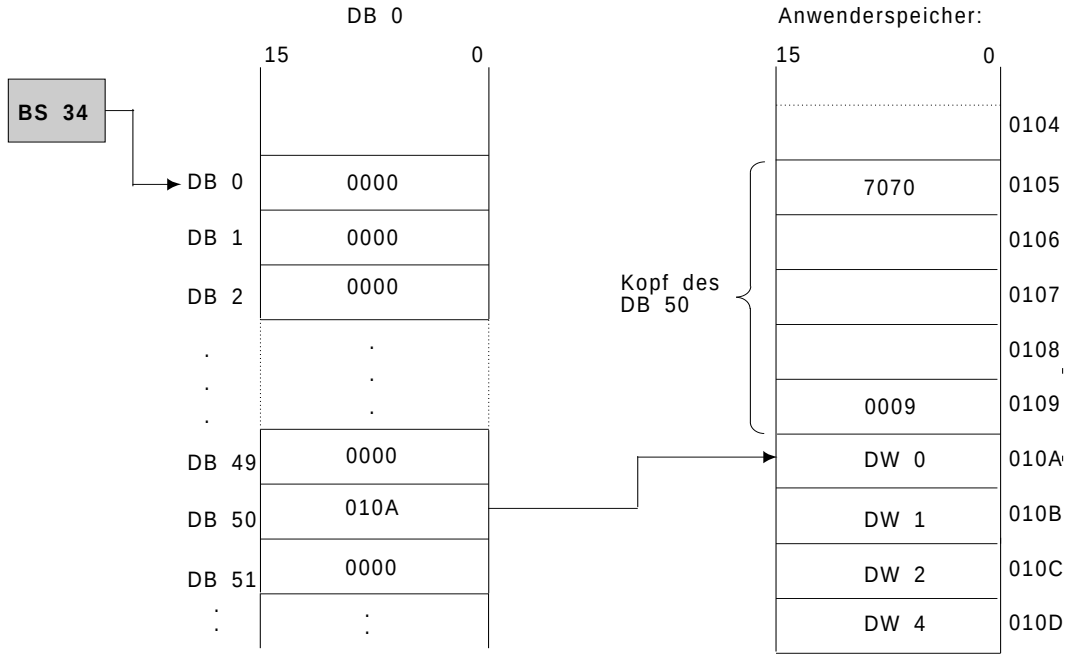


Bild 8-5 Anfangsadresse des DB 50

Ergebnis: AKKU-1-L: Länge des DB 50
AKKU-2-L: Anfangsadresse des DB 50

Fortsetzung: nächste Seite

Fortsetzung des Beispiels (Adresse un Länge von DB 50):

b) mit dem Sonderfunktionsorganisationsbaustein OB 181
"Datenbausteine (DB/DX) testen":

Der OB 181 (siehe Abschnitt 6.16) führt dieselbe Funktion aus wie unter Beispiel 2a beschrieben. Er testet jedoch zusätzlich, ob der Datenbaustein im Anwenderspeicher (RAM- oder EPROM-Modul) oder im DB-RAM liegt.

```
      :L    KY1,50     Datenbaustein DB 50
      :SPA  OB 181     "Datenbausteine (DB/DX) testen"
      :SPB  =NIVO      Sprung, wenn nicht vorhanden
      :SPM  =PROM      Sprung, wenn im EPROM-Modul
      :SPZ  =ANWE      Sprung, wenn im RAM-Modul
      :SPP  =DBRA      Sprung, wenn im DB-RAM
      :SPA  = FEHL        Sprung auf Fehlerbearbeitung
      :
NIVO  :                Datenbaustein nicht vorhanden
      :
      :BEA

PROM  :              Datenbaustein liegt im Anwenderspeicher
      :                 (EPROM-Modul)
      :BEA

ANWE  :              Datenbaustein liegt im Anwenderspeicher
      :                 (RAM-Modul)
      :BEA

DBRA  :              Datenbaustein liegt im DB-RAM
      :
      :BEA

FEHL:                Fehlerbearbeitung
      :
      :BE

      Ergebnis:  AKKU-1-L:  Anfangsadresse des DB 50
                 AKKU-2-L:  Länge des DB 50
                 VKE = 1, falls DB 50 nicht vorhanden
```

8.3.3 BA-/BB-Bereich

Der BA-Bereich ist ein 256 Wörter langer Bereich im internen System-RAM der CPU. Er belegt die Adressen E800H bis E8FFH.

Der BB-Bereich ist ein 256 Wörter langer Bereich im internen System-RAM der CPU. Er belegt die Adressen E900H bis E9FFH.

Der gesamte BA-Bereich (BA 0 bis BA 255) und der gesamte BB- Bereich (BB 0 bis BB 255) können vom Anwender für eigene Zwecke genutzt werden.

Der BA-/BB-Bereich wird nur bei URLÖSCHEN mit Nullen vorbesetzt!

8.3.4 BS-/BT-Bereich

BS- und BT-Bereich enthalten teils Informationen für den Systemprogrammierer, teils systeminterne Daten.

Der **BS-Bereich** ist ein 256 Wörter langer Bereich im internen System-RAM der CPU. Er belegt die Adressen EA00H bis EAFFH.

Warnung
Es dürfen ausschließlich die Systemdatenwörter BS 1, BS 60 bis BS 63, BS 133, BS 140 und BS 144 beschrieben werden:

- BS 60 bis BS 63 stehen für ihre eigenen Zwecke zur Verfügung.
- BS 1 und BS 133 haben eine festgelegte Bedeutung und beeinflussen die Programmbearbeitung. Sie dürfen nur mit gültigen Kennungen beschrieben werden!

Alle übrigen Systemdaten dürfen nur gelesen werden:

Ein Beschreiben dieser Systemdaten kann Rückwirkungen auf die Funktionsfähigkeit des Automatisierungsgerätes sowie angeschlossener Programmiergeräte zur Folge haben: Es können schwere Störungen entstehen, die Mensch und Maschine in Gefahr bringen können.

Der **BT-Bereich** ist ein 256 Wörter langer Bereich im internen System-RAM der CPU. Er belegt die Adressen EB00H bis EBFFH.

Den gesamten BT-Bereich (BT 0 bis BT 255) können Sie für Ihre eigenen Zwecke nutzen.

Der BS-/BT-Bereich wird nur bei URLÖSCHEN gelöscht.

Die Informationen einiger Systemdaten (über den internen Aufbau der CPU, den Ausgabestand der Software, die CPU-Kennung etc.) erhalten Sie außerdem über die Online-Funktion SYSTEM- PARAMETER.

Ergänzend zur Darstellung in den Bildern 8-6 und 8-7 werden nachfolgend die Bit-Belegungen einiger Systemdaten angegeben, die Sie über STEP-5-Operationen oder mit dem PG auswerten können (die Erklärung für die dort aufgeführten Abkürzungen entnehmen Sie bitte dem Abschnitt 5.3).

Belegung des BS-Bereiches

BS	Bezeichnung		Adr.
0	Unterbrechungsanzeigenwort (STOER.URS.)		EA00
1	Unterbrechungsanzeigenloeschwort (UALW)		EA01
2	Unterbrechungsanzeigensammelwort (UAMK)		EA02
3 4	Anlauf-Fehlererkennung-Anzeige		EA03 EA04
5	Stopp-Kennungen	Anlauf-Kennungen	EA05
6	Zyklus-Kennungen	Modul-Kennungen	EA06
7	Urlösch-Kennungen	Fehlerkennungen (H)	EA07
8	Fehlerkennungen (M)	Fehlerkennungen (L)	EA08
9		Akt. Ident.-Nr.	EA09
10	Basisadresse der Eingangssignalformer		EA0A
11	Basisadresse der Ausgangssignalformer		EA0B
12	Basisadresse Prozeßabbild der Eingänge		EA0C
13	Basisadresse Prozeßabbild der Ausgänge		EA0D
14	Basisadresse Merkerbereich		EA0E
15	Basisadresse Zeitbereich		EA0F
16	Basisadresse Zählerbereich		EA10
17	Basisadresse Bereich Anschaltung		EA11
18		AG-SW-Stand	EA12
19	Endadresse des Anwendermodulspeichers		EA13
20	Basisadresse Bereich System		EA14
21	Länge der DB-Adreßliste		EA15
22	Länge der SB-Adreßliste		EA16
23	Länge der PB-Adreßliste		EA17
24	Länge der FB-Adreßliste		EA18
25	Länge der OB-Adreßliste		EA19
26	Länge der FX-Adreßliste		EA1A
27	Länge der DX-Adreßliste		EA1B
28	Länge des Adrelisten-DBs (DB 0)		EA1C
29	Steckplatzkennung	CPU-Kennung 2 (Typ)	EA1D

: reserviert

Bild 8-6 Belegung des BS-Bereichs, 1. Teil

30	Länge der Bausteinkopf-Information	EA1E
31	CPU-Kennung 1 / SW-Stand PG-Ansch.	EA1F
32	Basisadresse DX-Adreßliste	EA20
33	Basisadresse FX-Adreßliste	EA21
34	Basisadresse DB-Adreßliste	EA22
35	Basisadresse SB-Adreßliste	EA23
36	Basisadresse PB-Adreßliste	EA24
37	Basisadresse FB-Adreßliste	EA25
38	Basisadresse OB-Adreßliste	EA26
39 – 54		EA27 – EA36
55	Zähler für 1 Std. (bis 3599 sec, hex.)	EA37
56 – 59	reserviert für Hantierungsbaustein	EA38 – EA3B
60 – 63	reserviert für Anwenderzwecke	EA3C – EA3F
64 – 79	reserviert für Systemprogramm	EA40 – EA4E
80	zusätzliche Fehlerkennung, wenn Bit FE-5 in BS 8 gesetzt	EA50
81 – 127	reserviert für Systemprogramm	EA51 – EA7F
128		EA80
129		EA81
130	Kennung "Regelung"	EA82
131	Anzeigenwort "Alarme gemeinsam sperren"	EA83
132	Anzeigenwort "Alarme gemeinsam verzögern"	EA84
133	Kennung "Prozessabbildaktualisierung"	EA85
134		EA86
135	Anzeigenwort "Weckalarme einzeln sperren"	EA87
136		EA88
137	Anzeigenwort "Weckalarme einzeln verzögern"	EA89
138 – 139		EA8A – EA8B
140	Anzeigenwort "Bausteine Schreiben und Löschen"	EA8C
141 – 143		EA8D – EA8F
144	alternatives Laden von Datenbausteinen	EA90
145 – 255		EA91 – EAFF

Bild 8-7 Belegung des BS-Bereichs, 2. Teil

8.3.5 Bitbelegung der Systemdatenwörter

Systemdatum BS 0 **Unterbrechungsanzeigenwort**

Adresse EA00H

Tabelle 8-2 Belegung BS 0 (Unterbrechungsanzeigenwort)

High-Byte	
Bit-Nr.	Belegung
15	NAU
14	PEU
13	BAU
12	MP-STP
11	ZYK
10	QVZ
9	ADF
8	STP
Low-Byte	
7	BCF
6	FE-3
5	LZF
4	REG
3	STUEB
2	STUEU
1	WECK
0	DOPP

Das Systemdatum BS 0 entspricht der STOERUNGSURSACHE im USTACK. Tritt bei der Programmbearbeitung z. B ein Laufzeitfehler auf, wird das Bit Nr. 5 gesetzt. Ist die Programmbearbeitungsebene LZF vollständig bearbeitet, wird das Bit Nr. 5 rückgesetzt.

Systemdatum BS 1

Unterbrechungsanzeigen-Löschwort UALW

Adresse: EA01H

BS 1: Aktive Schnittstelle, für Anwender freigegeben!

Durch Setzen von Bit Nr. 9 bzw. Bit Nr. 10 des UALW erreichen Sie, daß der **nächstfolgende** ADF bzw. QVZ ignoriert wird und die laufende Programmbearbeitung dadurch nicht beeinflußt wird. Nach Auftreten eines QVZ bzw. ADF setzt das Systemprogramm das betreffende Bit zurück.

Tabelle 8-3 Belegung BS 1 (Unterbrechungsanzeigen-Löschwort)

High-Byte	
Bit-Nr.	Belegung
15	nicht belegt
14	
13	
12	
11	
10	QVZ
9	ADF
8	nicht belegt
Low-Byte	
7	nicht belegt
6	
5	
4	
3	
2	
1	
0	

Jede Programmbearbeitungsebene hat ihr **eigenes UALW**!

Beispiel zum UALW

```
In diesem Beispiel wird getestet, ob unter einer bestimmten Peripheriadresse eine
Baugruppe ansprechbar ist. Ist die Baugruppe nicht vorhanden, wird mit Hilfe des
UALW ein Quittungsverzug verhindert und ein für diesen Fall vorgesehenes
Programm bearbeitet. Ebenso wird getestet, ob eine bestimmte Peripherieadresse im
DB 1 eingetragen ist. Falls nicht, wird mit Hilfe des UALW ein Adressierfehler
verhindert und ein spezielles Programm bearbeitet.


FB 201
NAME:L

     :SPA  FB  10
NAME:PERITEST              Testen, ob unter der Peripherieadresse 128
PADR  :  PB 128            eine Baugruppe ansprechbar ist
MASK  :  KM 00000100 00000000
     :SPN  =M001
     :..                   Dieser Programmteil wird bearbeitet, falls
     :..                   Baugruppe nicht ansprechbar ist
     :..
M001  :
     :SPA  FB 10
NAME:PERITEST              Testen, ob im DB 1 eine Baugruppe
PADR  :  AB 4                 mit der Peripherieadresse 4
MASK  :  KM 00000010 00000000  eingetragen ist
     :SPN  =M002
     :..                   Dieser Programmteil wird bearbeitet,
     :..                      falls Peripherieadresse nicht
     :..                      eingetragen ist
M002  :
     :BE



FB 10
NAME: PERITEST
BEZ   :PADR E/A/D/B/T/Z:   E BI/BY/W/D:      BY
BEZ   :MASK E/A/D/B/T/Z:   D KM/KH/KY/KC/KF/KT/KZ/KG: KM

     :L    BS 1            UALW laden und sichern
     :T    BS 60
     :LW   =MASK           QVZ- bzw. ADF-Bit setzen
     :OW
     :T    BS 1            UALW zurueckschreiben
     :L    =PADR           Einzel-Peripheriezugriff bzw. Zugriff auf
     :                        das Prozeßabbild
     :L    BS 1
     :LW   =MASK           QVZ- bzw. ADF-Bit maskieren
     :UW
     :L    BS 60           Altes UALW zurueckschreiben, damit der
     :T    BS 1               nächste QVZ bzw. ADF erkannt wird
     :TAK
     :BE
```

Systemdatum BS 2

Unterbrechungsanzeigen-Sammelwort UAMK

Adresse: EA02H

Die 16 Bits des Unterbrechungsanzeigen-Sammelworts entsprechen den unter STOERUNGSURSACHE im USTACK aufgeführten möglichen Fehlerursachen.

Bei Auftreten eines bestimmten Fehlers wird das dazugehörige Bit gesetzt.

Tabelle 8-4 Belegung BS 2 (Unterbrechungsanzeigen-Sammelwort)

High-Byte	
Bit-Nr.	Belegung
15	NAU
14	PEU
13	BAU
12	MP-STP
11	ZYK
10	QVZ
9	ADF
8	STP
Low-Byte	
7	BCF
6	FE-3
5	LZF
4	REG
3	STUEB
2	STUEU
1	WECK
0	DOPP

Das Unterbrechungsanzeigen-Sammelwort (UAMK im USTACK) darf nur gelesen werden!

Beispiel zum UAMK

Geht die CPU aufgrund eines Adressierfehlers (ADF) in STOP, so wird im UAMK Bit Nr. 9 gesetzt. Wenn nun bei der Bearbeitung des ADF ein Befehlscodefehler (BCF) auftritt, wird im UAMK außerdem Bit Nr. 7 gesetzt.

Inhalt des UAMK (binär):	00000010 10000000
Darstellung (hexadezimal) im USTACK:	0280

Während im USTACK unter STOERUNGSURSACHE nur der jeweils zuletzt aufgetretene Fehler angekreuzt ist, sind im UAMK alle bis dahin aufgetretenen Fehler **aufsummiert** (USTACK Tiefe 05: im UAMK sind 5 Bits gesetzt). Durch Umwandlung des Hexadezimalcodes in den Binärcode läßt sich der Inhalt des UAMK auswerten. Auf diese Weise können Sie feststellen, welche Fehler den Stoppzustand verursacht haben.

Die Fehler-Bits werden rückgesetzt, sobald die entsprechende Fehler-Programmbearbeitungsebene vollständig bearbeitet ist und damit verlassen wird.

Unterbrechungsanzeigen von Fehlern, denen keine Programmbearbeitungsebene zugeordnet ist (z. B. NAU, PEU, STUEB usw.), werden im ANLAUF gelöscht.

Systemdatum BS 5

STOP- und ANLAUF-Kennungen

Adresse: EA05H

Die Kennungen entsprechen den Steuerbits in Zeile 1 und 2 des USTACK.

Tabelle 8-5 Belegung BS 5 (STOP- und ANLAUF-Kennungen)

High-Byte: STOP-Kennungen	
Bit-Nr.	Belegung
15	PRI-STP
14	nicht belegt
13	FE-STP
12	BARB-END
11	PG-STP
10	STP-SCH
9	STP-BEF
8	MP-STP
Low-Byte: ANLAUF-Kennungen	
7	ANL
6	nicht belegt
5	NEUST
4	MWA
3	AWA
2	nicht belegt
1	NEU-ZUL
0	MWA-ZUL

Systemdatum BS 6

ZYKLUS- und Modul-/MPL-Kennungen

Adresse: EA06H

Die Kennungen entsprechen den Steuerbits in Zeile 3 und 4 des USTACK.

Tabelle 8-6 Belegung BS 6 (ZYKLUS- und Modul-/MPL-Kennungen)

High-Byte: ZYKLUS-Kennungen	
Bit-Nr.	Belegung
15	RUN
14	nicht belegt
13	EIN-PROZ
12	BARB
11	OB1-GEL
10	FB0-GEL
9	OB-PROZA
8	OB-WECKA
Low-Byte: Modul-/MPL-Kennungen	
7	32KW-RAM
6	16KW-RAM
5	8KW-RAM
4	EPROM
3	KM-AUS
2	KM-EIN
1	DIG-EIN
0	DIG-AUS

BS 7 **URLÖSCH-Kennungen/Fehlerkennungen Initialisieren**

Adresse: EA07H

Die Kennungen entsprechen den Steuerbits in Zeile 5 und 6 des USTACK.

Tabelle 8-7 Belegung BS 7 (URLÖSCH-/Fehler-Kennungen Initialisieren)

High-Byte: URLÖSCH-Kennungen	
Bit-Nr.	Belegung
15	URGELOE
14	URL-IA
13	STP-VER
12	ANL-ABB
11	UA-PG
10	UA-SYS
9	UA-PRFE
8	UA-SCH
Low-Byte: Fehlerkennungen Initialisieren	
7	DX0-FE
6	nicht belegt
5	MOD-FE
4	RAM-FE
3	DB0-FE
2	DB1-FE
1	DB2-FE
0	KOR-FE

BS 8 **Fehlerkennungen HW/SW**

Adresse: EA08H

Die Kennungen entsprechen den Steuerbits in Zeile 7 und 8 des USTACK.

Tabelle 8-8 Belegung BS 8 (Fehlerkennungen HW/SW)

Bit-Nr.	High-Byte: Fehlerkennungen HW
15	NAU
14	PEU
13	BAU
12	STUE-FE
11	ZYK
10	QVZ
9	ADF
8	WECK-FE
Bit-Nr.	**Low-Byte: Fehlerkennungen SW**
7	BCF
6	nicht belegt
5	FE-5
4	Power-down-Fehler
3	FE-3
2	LZF
1	REG-FE
0	DOPP-FE

BS 29 **Steckplatzkennung/CPU-/AG-Typ**

Adresse: EA1DH

Tabelle 8-9 Belegung BS 29 (Steckplatz-Kennung/CPU-/AG-Typ)

Bit-Nr.	High-Byte: Fehlerkennungen HW
15	nicht belegt
14	
13	
12	
11	CPU Nr. 4
10	CPU Nr. 3
9	CPU Nr. 2
8	CPU Nr. 1
Bit-Nr.	**Low-Byte: Fehlerkennungen SW**
7	CPU-Typ
6	
5	
4	
3	AG-Typ
2	
1	
0	

BS 29 (high) Aktive Schnittstelle, wird von den Hantierungsbausteinen und bei der Mehrprozessorkommunikation sowie vom OB 218 und den Operationen SES und SEF benutzt.

BS 29 (low) CPU-Typ: 1011 CPU 928B

AG-Typ: 0111 AG S5-135U

BS 80

Adresse: EA50H (high und low)

Hier steht eine zusätzliche Information zur Spezifizierung des aufgetretenen Fehlers, wenn das Bit Nr. 5 in BS 8 vom System gesetzt bzw. wenn das Steuerbit FE-5 in der USTACK-Ausgabe angekreuzt ist:

Kennung in BS 80	Fehlerursache
2460H	Ready-Signal auf S5-Bus steht dauernd an

BS 130

Adresse EA82 (low)

Das Systemdatum BS 130 zeigt Ihnen folgende Zustände der Programmbearbeitungsebene "Regelung" an:

Bit-Nr. 0 = 0 : Programmbearbeitungsebene "Regelung" aktiviert

Bit-Nr. 0 = 1 : Programmbearbeitungsebene "Regelung" unterdrückt

Vor Aufruf eines Anlauf-Organisationsbausteins (OB 20, 21 oder 22) wertet das Systemprogramm den Datenbaustein DB 2 aus (falls vorhanden). Je nach Ergebnis der Auswertung wird das BS 130 vom Systemprogramm gesetzt bzw. rückgesetzt. Danach ruft das Systemprogramm einen Anlauf-OB auf.
Ist das BS 130 (LOW) rückgesetzt, wird die Reglerbearbeitung im zyklischen Betrieb entsprechend der Reglerliste im DB 2 durchgeführt.

BS 131

Anzeigenwort "Alarme gemeinsam sperren": siehe OB 120 (Abschnitt 6.5)
Adresse EA83 (low)

Das Systemdatum BS 131 zeigt Ihnen folgende Zustände der Programmbearbeitungsebenen "Alarmbearbeitung" an:

Tabelle 8-10 Belegung BS 131 (Alarme gemeinsam sperren)

Bit-Nr.	Low-Byte: Alarme gemeinsam sperren
7	0
6	0
5	0
4	0
3	Verzögerungsalarm
2	Prozeßalarm
1	uhrzeitgesteuerter Weckalarm
0	Weckalarme mit festem Zeitraster

Bit = ’1’ bedeutet: Alarm(e) ist (sind) gesperrt.

BS 132

Anzeigenwort "Alarme gemeinsam verzögern": siehe OB 122 (Abschnitt 6.7)
Adresse EA84 (low)

Das Systemdatum BS 131 zeigt Ihnen folgende Zustände der Programmbearbeitungsebenen "Alarmbearbeitung" an:

Tabelle 8-11 Belegung BS 132 (Alarme gemeinsam verzögern)

Bit-Nr.	Low-Byte: Alarme gemeinsam verzögern
7	0
6	0
5	0
4	0
3	Verzögerungsalarm
2	Prozeßalarm
1	uhrzeitgesteuerter Weckalarm
0	Weckalarme mit festem Zeitraster

Bit = ’1’ bedeutet: Alarm(e) ist (sind) verzögert.

BS 133 **Prozeßbildaktualisierung**

Adresse EA85 (low)

Tabelle 8-12 Belegung BS 133 (Prozeßbildaktualisierung)

Bit-Nr.	Low-Byte: Prozeßbildaktualisierung
7	nicht belegt
6	
5	
4	
3	KM-AUS
2	KM-EIN
1	DIGH-EIN
0	DIG-AUS

Bit-Nr. 0 = 0: Nächstes Prozeßabbild der digitalen Ausgänge wird ausgegeben.

Bit-Nr. 0 = 1: Nächste Prozeßabbildaktualisierung der digitalen Ausgänge wird unterdrückt.

Bit-Nr. 1 = 0: Nächstes Prozeßabbild der digitalen Eingänge wird eingelesen.

Bit-Nr. 1 = 1: Nächste Prozeßabbildaktualisierung der digitalen Eingänge wird unterdrückt.

Bit-Nr. 2 = 0: Nächstes Prozeßabbild der Koppelmerkereingänge wird eingelesen.

Bit-Nr. 2 = 1: Nächste Prozeßabbildaktualisierung der Koppelmerkereingänge wird unterdrückt.

Bit-Nr. 3 = 0: Nächstes Prozeßabbild der Koppelmerkerausgänge wird ausgegeben

Bit-Nr. 3 = 1: nächste Prozeßabbildaktualisierung der Koppelmerkerausgänge wird unterdrückt

Hinweis
Jedes Bit verhindert, falls es gesetzt ist, die Prozeßabbildaktualisierung **einmal**, anschließend wird es vom Systemprogramm **sofort wieder auf '0' gesetzt**.

BS 135

Anzeigenwort "Weckalarme einzeln sperren": siehe OB 121 (Abschnitt 6.6)
Adresse EA87

Das Systemdatum BS 135 zeigt Ihnen folgende Zustände der Programmbearbeitungsebenen "Weckalarmbearbeitung" an:

Tabelle 8-13 Belegung BS 135 (Weckalarme einzeln sperren)

Bit-Nr.	High-Byte: Weckalarme einzeln sperren
15	0
14	0
13	0
12	0
11	Weckalarm 5 s (OB 18)
10	Weckalarm 2 s (OB 17)
9	Weckalarm 1 s (OB 16)
8	Weckalarm 500 ms (OB 15)
Bit-Nr.	**Low-Byte: Weckalarme einzeln sperren**
7	Weckalarm 200 ms (OB 14)
6	Weckalarm 100 ms (OB 13)
5	Weckalarm 50 ms (OB 12)
4	Weckalarm 20 ms (OB 11)
3	Weckalarm 10 ms (OB 10)
2	0
1	0
0	0

Bit = ’1’ bedeutet: dieser Weckalarm ist gesperrt.

BS 137

Anzeigenwort "Weckalarme einzeln verzögern": siehe OB 123 (Abschnitt 6.8)
Adresse EA89

Das Systemdatum BS 137 zeigt Ihnen folgende Zustände der Programmbearbeitungsebenen "Weckalarmbearbeitung" an:

Tabelle 8-14 Belegung BS 137 (Weckalarme einzeln verzögern)

Bit-Nr.	High-Byte: Weckalarme einzeln verzögern
15	0
14	0
13	0
12	0
11	Weckalarm 5 s (OB 18)
10	Weckalarm 2 s (OB 17)
9	Weckalarm 1 s (OB 16)
8	Weckalarm 500 ms (OB 15)
Bit-Nr.	**Low-Byte: Weckalarme einzeln verzögern**
7	Weckalarm 200 ms (OB 14)
6	Weckalarm 100 ms (OB 13)
5	Weckalarm 50 ms (OB 12)
4	Weckalarm 20 ms (OB 11)
3	Weckalarm 10 ms (OB 10)
2	0
1	0
0	0

Bit = ’1’ bedeutet: dieser Weckalarm ist verzögert.

BS 140

Anzeigenwort "Bausteine Schreiben und Lesen"

Adresse EA8C

Das Systemdatum BS 140 zeigt Ihnen an, ob seit dem letzten URLÖSCHEN der CPU bzw. seit dem letzten Löschen des Systemdatums BS 140 Bausteine überschrieben, neu geladen oder gelöscht wurden. Für jeden Baustein werden die Bits für Veränderung und Bausteintyp dazugeodert. Vor einem neuen Überwachungsabschnitt muß das Systemdatum BS 140 gelöscht werden. Das BS 140 wird beim URLÖSCHEN ebenfalls gelöscht.

Tabelle 8-15 Belegung BS 140 (Schreib-/Lese-Kennungen)

Bit-Nr.	**High-Byte: Schreib-/Lese-Kennungen**
15	Baustein gelöscht
14	Baustein geladen
13	Baustein überschrieben
12	nicht belegt
11	
10	
9	
8	
Bit-Nr.	**Low-Byte: Schreib-/Lese-Kennungen**
7	nicht belegt
6	DX
5	DB
4	FX
3	FB
2	SB
1	PB
0	OB

BS 144 **"alternatives Laden von Datenbausteinen in das DB-RAM"**

Adresse EA90

Standardmäßig werden bei der CPU 928B alle Bausteine vom PG zuerst in das Anwenderspeichermodul geladen. Erst wenn dort kein Platz mehr frei ist, werden **Datenbausteine** (DX, DX) und nur diese im DB-RAM abgelegt.

Über Bit Nr. 0 des Systemdatenwortes BS 144 können Sie die Reihenfolge beim Laden der Datenbausteine beeinflussen:

Bit 0 = 0: Voreinstellung "Standard-Verhalten":
Die Datenbausteine werden zuerst in das Anwenderspeichermodul geladen. Erst wenn dort kein Platz mehr frei ist, werden sie in das DB-RAM geladen.

Bit 0 = 1: Die Datenbausteine werden zuerst in das DB-RAM geladen. Erst wenn dort kein Platz mehr frei ist, werden sie in das Anwenderspeichermodul geladen.

Die übrigen Bits von BS 144 sind nicht belegt.

Hinweis
Codebausteine werden unabhängig von der Einstellung in BS 144 **nur in den Anwenderspeicher** geladen.

Die Einstellung in BS 144 hat keinen Einfluß auf Operationen und Sonderfunktions-OBs zum Erzeugen und Umladen von Bausteinen.

Speicherzugriffe über absolute Adressen

9

Inhalt von Kapitel 9

9 Speicherzugriffe über absolute Adressen

In diesem Kapitel erfahren Sie, wie Sie mit Hilfe von STEP-5-Operationen und speziellen STEP-5-Registern Daten in bestimmten Speicherbereichen über Absolutadressen ansprechen können.

9.1 Einführung

Die Programmiersprache STEP 5 enthält Operationen, mit denen Zugriffe auf den gesamten Adreßraum möglich sind. Diese Operationen gehören zu den "Systemoperationen".

Warnung
Bei einer nicht sachgerechten Anwendung dieser Operationen können STEP-5-Bausteine und Systemdaten überschrieben werden. Dies kann unerwünschte Betriebszustände zur Folge haben. Operationen, die mit absoluten Adressen arbeiten, sollten deshalb nur von Anwendern mit sehr guten Systemkenntnissen benutzt werden**.**

Lokaler Speicher

Als lokaler Speicher wird der Speicherbereich bezeichnet, der auf jeder CPU vorhanden ist (Anwenderspeicher, DB-RAM, BA-, BB-, BS-, BT-Bereich, Zähler, Zeiten, Merker, Prozeßabbild).

Globaler Speicher

Der globale Speicher ist nur einmal für alle CPUs vorhanden und wird über den S5-Bus adressiert.

Speicherorganisation

Speicherbereiche sind **byteweise** oder **wortweise** organisiert.

- Byteweise Organisation: Jede Adresse adressiert ein Byte.
- Wortweise Organisation: Jede Adresse adressiert ein 16-bit-Wort (= 2 Bytes).

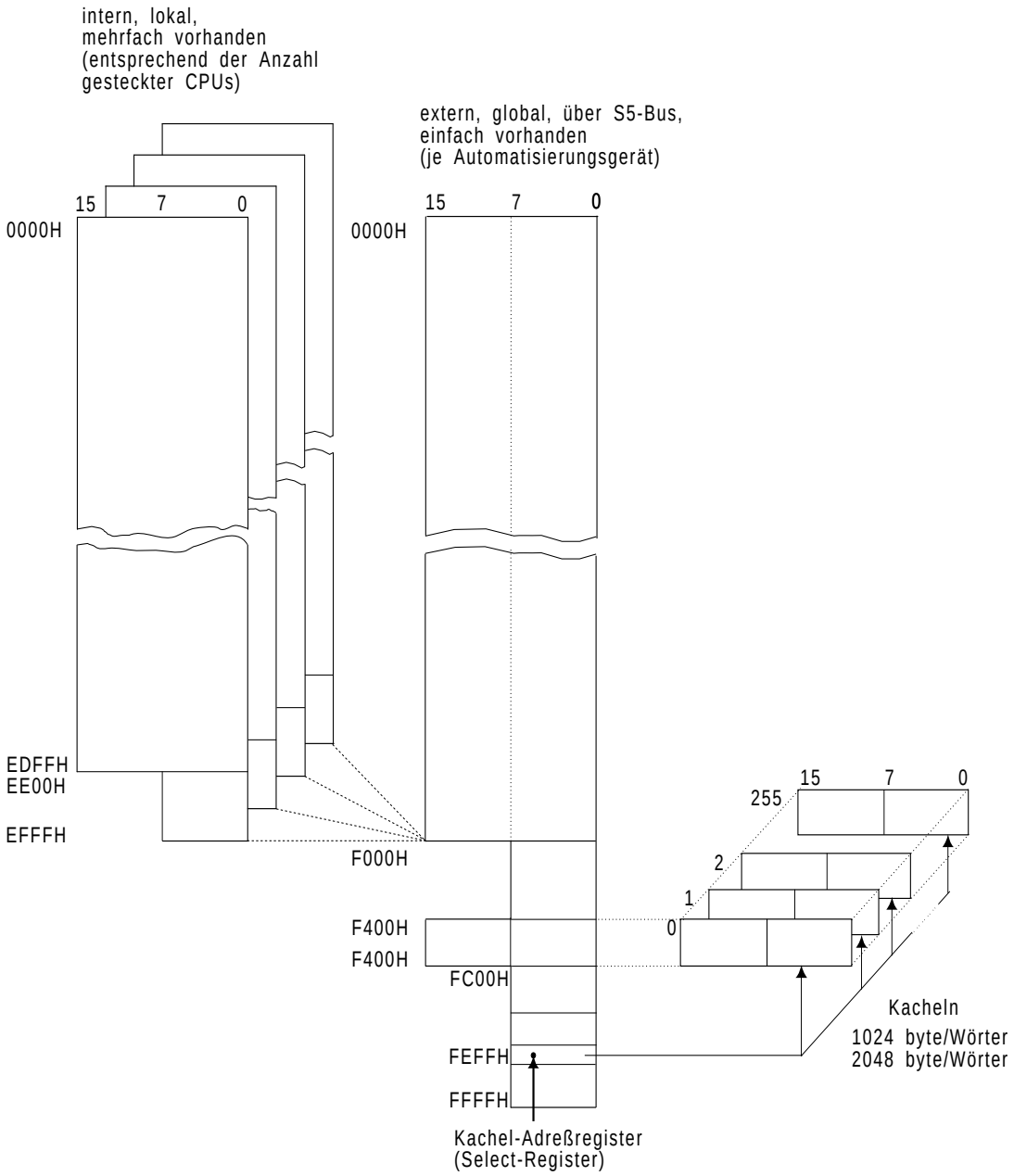


Bild 9-1 Globaler und lokaler Speicher

Speicherzugriffe

Die folgenden Operationen ermöglichen Zugriffe auf lokale bzw. globale Speicherbereiche über absolute Adressen (siehe auch Bild 9-2).

Zugriffe auf den Lokal- und Globalbereich

Sie können sowohl auf den Lokal- als auch auf den Globalbereich zugreifen:

- auf Lokalbereich (Adressen 0000H bis EFFFH) und den byteweise organisierten Teil des Globalbereichs (Adressen F000H bis F3FFH, FC00H bis FFFFH) mit:

  LIR, TIR, TNB, TNW,

- auf den wortweise organisierten Teil des Lokalbeeiches (Adressen 0000H bis E3FFH sowie E800H bis EDFFH) mit:

  LRW, TRW, LRD, TRD.

Zugriffe nur auf den Globalbereich

Sie können auf folgende Teile des Globalbereichs zugreifen:

- auf den byteweise organisierten Teil des Globalbereiches (Adressen 0000H bis EFFFH) mit:

  LB GB, LB GW, LB GD, TB GB, TB GW, TB GD, TSG,

- auf den wortweise organisierten Teil des Globalbereichs (Adressen 0000H bis EFFFH) mit:

  LW GW, LW GD, TW GW, TW GD, TSG .

Zugriffe auf den Kachelbereich

Sie können auf folgende Teile des Kachelbereichs zugreifen:

- auf den byteweise organisierten Teil des Globalbereichs (Adressen F400H bis FBFFH, = Kachelbereich):

  LB CB, LB CW, LB CD, TB CB, TB CW, TB CD, TSC,

- auf den wortweise organisierten Teil des Globalbereichs (Adressen F400H bis FBFFH, = Kachelbereich):

  LW CW, LW CD, TW CW, TW CD, TSC

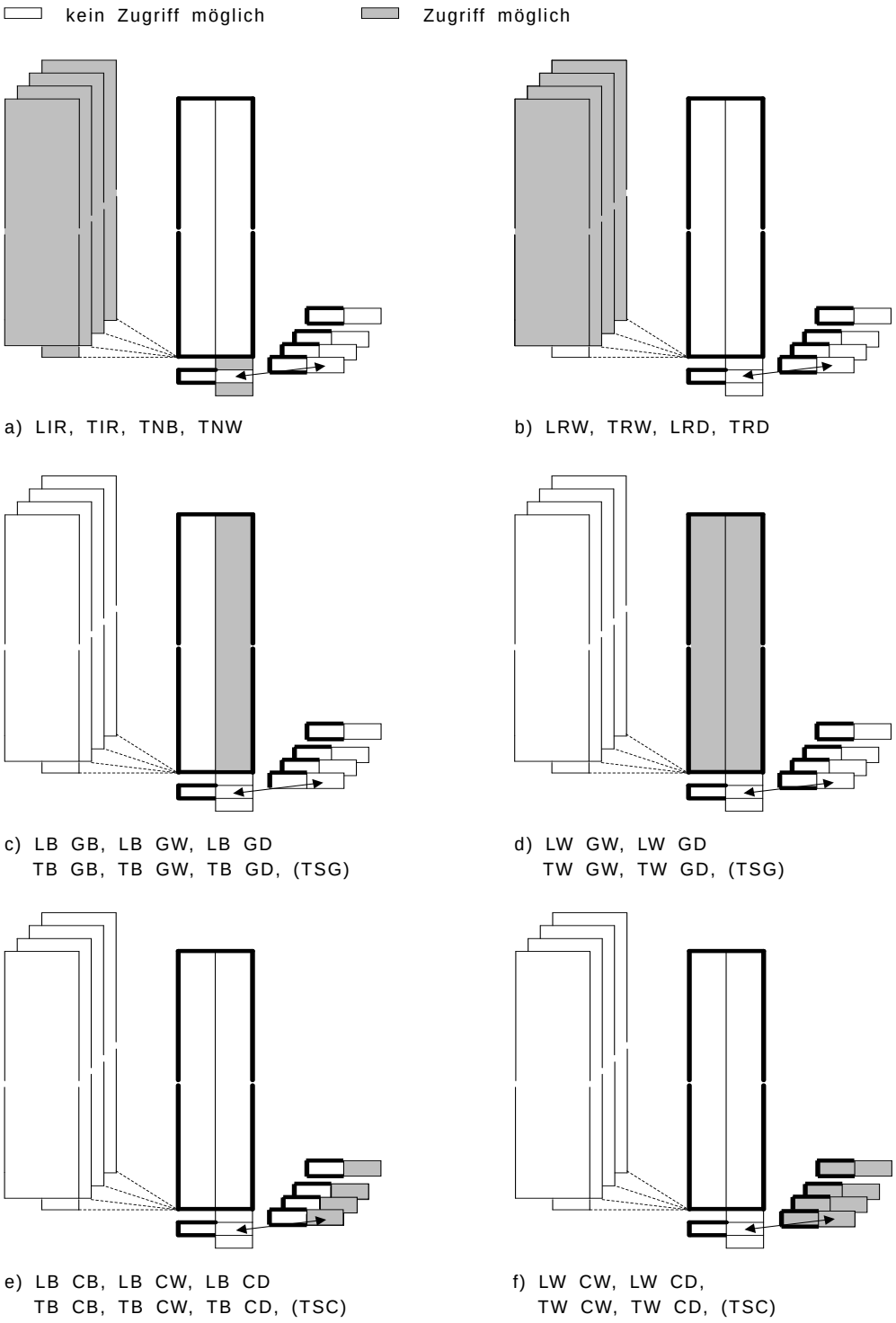


Bild 9-2 Zugriffe auf lokale bzw. gobale Speicherbereiche über absolute Adressen (siehe auch Bild 9-1)

9.2 Speicherzugriffe über Adresse in AKKU 1

Anwendung

Die Register sind Betriebsmittel der CPU zum Bearbeiten des STEP-5-Programms. Jedes Register ist 16 bit breit. Mit den Systemoperationen LIR (Lade indirekt Register) und TIR (Transferiere indirekt Register) können Sie auf die Inhalte der Register zugreifen.

Operationen

Tabelle 9-1 Operationen für indirekte Speicherzugriffe über Register

Operation	Operand	Funktion
LIR	Register-Nr. 0 bis 15	das **angegebene Register** mit dem Inhalt eines durch AKKU-1- adressierten Speicherwortes laden
TIR	Register-Nr. 0 bis 15	den Inhalt des **angegebenen Registers** in das durch AKKU-1-L adressierten Speicherwortes laden

Das Speicherwort liegt entweder im Lokalbereich (Adressen 0000H bis EFFFH) oder im byteweise organiserten Teil des Globalbeeichs (Adressen F000H bis F3FFH, FC00H bis FFFFH).

Auf den folgenden Seiten erfahren Sie, **welche Register** Sie bei den Operationen verwenden können.

Einige Beispiele erläutern Ihnen dann, **wie** Sie die Operationen anwenden.

9.2.1 LIR/TIR: 16-bit-Register indirekt laden/transferieren

Die folgende Tabelle zeigt Ihnen, welche Registernummern Sie bei der CPU 928B für die Operationen LIR und TIR verwenden dürfen und wie diese belegt sind.

Tabelle 9-2 16-bit-Register für LIR/TIR

Register-Nr.	Registerbelegung (je 16 bit breit)	
0	AKKU-1-H	(linkes Wort von AKKU1, Bit 16 - 31)
1	AKKU-1-L	(rechtes Wort von AKKU1, Bit 0 - 15)[1)]
2	AKKU-2-H	
3	AKKU-2-L	
6	DBA	(Datenbaustein-Anfangsadresse)
8	DBL	(Datenbaustein-Länge)
9	AKKU-3-H	
10	AKKU-3-L	
11	AKKU-4-H	
12	AKKU-4-L	
15	SAZ	(STEP-Adreßzähler)

1) Soll über AKKU-1-L der Inhalt der adressierten Speicherzelle in das Register ’1’ geladen werden, so wird dadurch die in AKKU-1-L hinterlegte Adresse überschrieben.

Die Register 4, 5, 7, 13und 14 sind bei der CPU 928B nicht vorhanden. LIR/TIR-Operationen mit diesen Registernummern werden wie eine **Nulloperation** (NOP) behandelt..

LIR und TIR auf den Kachelbereich

Die Befehle LIR und TIR sind im Mehrprozessor-Automatisierungsgerät AG S5-135U **nicht** für den Zugriff auf den Kachelbereich (Adressen F400H bis FBFFH) geeignet. Verwenden Sie stattdessen die Befehle aus Abschnitt 9.4.4 "Zugriff auf den Kachelspeicher" oder die Sonderfunktionen aus Abschnit 6.21 "Kachelzugriffe".

LIR/TIR auf 8-bit-Speicherbereiche

Wird mit LIR/TIR auf Speicherbereiche zugegriffen, die nur 8 bit breit sind (Speicheradressen von E400H bis E7FFH sowie ≥ EE00H, so beachten Sie, daß

- bei TIR nur das Low-Byte des Registers übertragen wird (das High-Byte des Registers geht verloren)

  und

- bei LIR das High-Byte des Registers mit **FFH** überschrieben wird.

Die Bilder 9-3 und 9-4 zeigen Ihnen den Unterschied bei Zugriffen mit LIR/TIR auf wort-und byteweise orientierte Speicherbereiche:

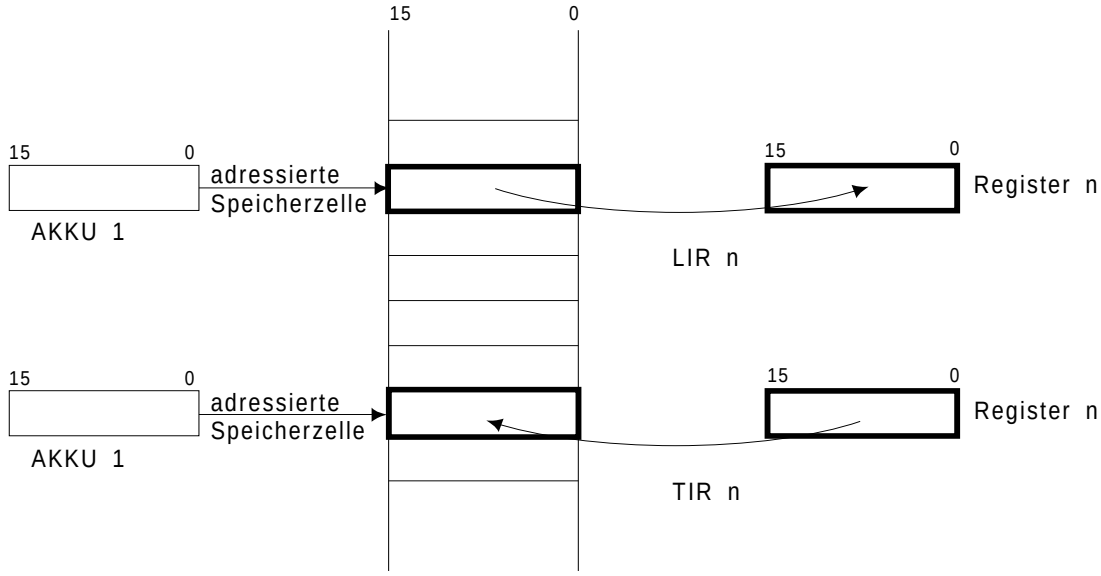


Bild 9-3 LIR/TIR auf 16-bit-Speicherbereiche (wortweise orientiert)

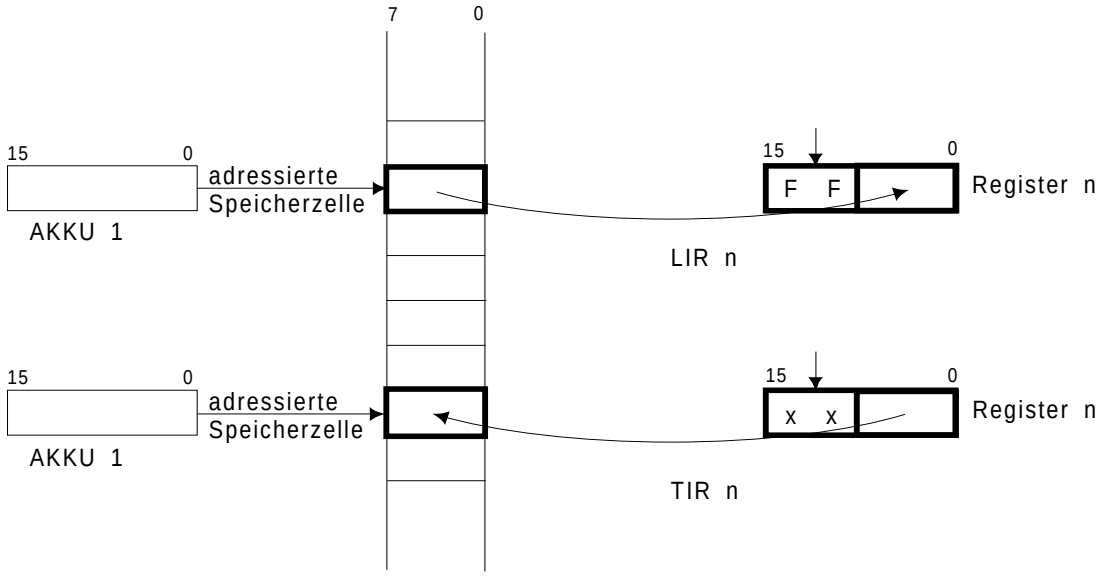


Bild 9-4 LIR/TIR auf 8-bit-Speicherbereiche (byteweise orientiert)

Register 0 bis 3 und 9 bis 12: AKKU 1, 2, 3 und 4

Die Akkumulatoren werden von der CPU bei der Programmbearbeitung als Zwischenspeicher verwendet. Mit den Operationen TIR und LIR können Sie die Inhalte der AKKUs in absolut adressierte Speicherzellen transferieren bzw. die Inhalte absolut adressierter Speicherzellen in die AKKUs laden. Die Absolutadresse der Speicherzelle steht jeweils im AKKU-1-L.

Beispiele

```
Der Inhalt der Speicherzelle mit der Adresse A000 wird ins Merkerwort MW 100
geladen.

   :L    KH A000       Adresse A000 der Speicherzelle in AKKU 1 laden
   :LIR  1             Inhalt der durch AKKU 1 adressierten Speicherzelle
   :                   ins Register 1 = AKKU-1-L laden
   :T    MW 100        Inhalt der Adresse A000 im Merkerwort MW 100 ablegen
   :BE
```

```
Den Inhalt des Merkerwortes 200 wird in die Speicherzelle mit der Adresse A000
transferiert.

   :L    MW 200        Merkerwort MW 200 in den AKKU 1 laden
   :L    KH A000       Adresse A000, auf die transferiert werden soll, in
   :                   den AKKU 1 laden (Merkerwort MW 200 nach AKKU 2)
   :TIR  3             Inhalt von Register 3 = AKKU-2-L in die durch AKKU 1
   :                   adressierte Speicherzelle transferieren
   :BE
```


Register 6: DBA (Datenbaustein-Anfangsadresse)

Beim Aufschlagen eines Datenbausteins mit den Befehlen A DB und AX DX wird das Register 6 mit der Adresse des DW 0 im aufgeschlagenen Datenbaustein geladen. Diese Adresse ist in der Bausteinadreßliste im DB 0 enthalten.
Das DBA-Register wird vor jedem Aufruf des OB 1 oder FB 0 gleich ’0’ gesetzt.

Das DBA-Register bleibt **erhalten**, wenn

- durch eine Sprunganweisung (SPA/SPB) die Programmbearbeitung in einem anderen Baustein fortgesetzt wird

  oder

- eine andere Programmbearbeitungsebene eingeschachtelt wird.

Es **ändert** sich, wenn

- ein anderer Datenbaustein aufgeschlagen wird

  oder

- ein Rücksprung in einen übergeordneten Baustein erfolgt nachdem im aufgerufenen Baustein ein neuer Datenbaustein aufgeschlagen wurde (siehe auch Abschnitt 2.4.3).

Hinweis
Im USTACK ist die im DBA-Register eingetragene Adresse unter 'DB-ADR' angegeben.

Zugriffe auf Datenwörter erfolgen normalerweise mit den STEP-5-Operationenen L/T DW, L/T DR, L/T DL, L/T DD, U/O/UN/ON/=/S/R Dx.y. Diese sind jedoch nur bis Datenwort DW 255 möglich. Durch Manipulieren des DBA-Registers können Sie mit diesen Befehlen auch auf Datenwörter > 255 zugreifen. Sie können dies auch mit Hilfe des Sonderfunktions-Organisationsbausteins OB 180 (siehe Kapitel 6.15) erreichen.

Beispiele

Beispiel 1: Wirkung der Operation **"AX DX 17"** auf das DBA-Register:

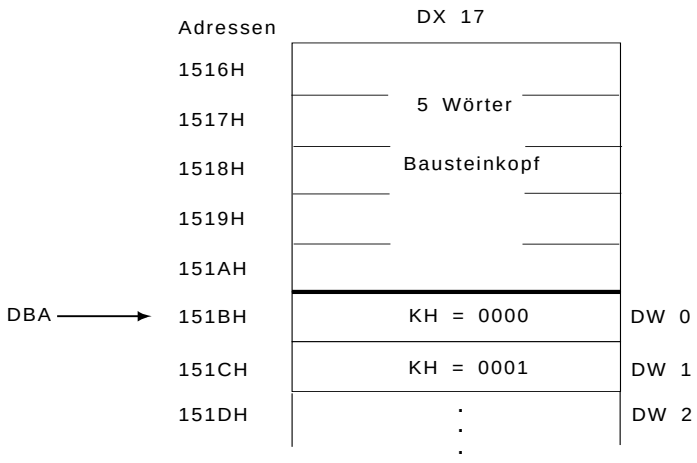


Bild 9-5 Verwendung des DBA-Registers

Bei Aufruf des DX 17 wird im DBA-Register die Adresse des Speicherwortes, in welchem das Datenwort DW 0 hinterlegt ist, im Beispiel: DBA = **4152H,** eingetragen.

Hinweis: Im USTACK ist die im DBA-Register eingetragene Adresse unter 'DB-ADR' angegeben.

Beispiel 2: Durch Ändern von Register 6 wird das Datenwort DW 300 des Datenbausteins DB 100 geladen.

```
FB 7
NAME: LIR/TIR6

     :L    BS 34      Anfangsadresse der DB-Adressliste plus 100
     :ADD  BF +100       ergibt Adresslisteneintrag des DB 100
     :LIR  1          Anfangsadresse des DB 100 (DW 0) nach AKKU 1
     :ADD  KF +200    Adresse des DW 200 im DB 100 im Systemdaten-
     :T    BS 62         wort BS 62 ablegen
     :L    BS 20      Basisadresse Systemdaten laden
     :ADD  KF +62     Adresse des BS 62 in AKKU 1 laden
     :LIR  6          DBA-Register mit dem Inhalt der Adresse des BS 62
     :                   laden, d. h. der Datenbausteinanfang wird auf
     :                   DW 200 gesetzt
     :L    DW 100     DW (200 + 100) = DW 300 laden
     :T    MW 100     DW 300 im Merkerwort MW 100 ablegen
     :BE
```

Beispiel 3: Verändern von DBA- und DBL-Register.

```
FB7
NAME:OB180

     :A    DB 100     DBA- und DBL-Register mit den Werten des
     :L    KF 200        DB 100 laden und mit Hilfe des OB 180 das
     :SPA  OB 180        DBA-Register um 200 erhoehen und das DBL-
     :                   Register um 200 vermindern
     :SPB  =FEHL      Fehlerausgang, falls der DB 100 weniger oder gleich
     :                   200 Datenwoerter enthaelt
     :L    DW 100     DW 300 laden und
     :T    MW 100        im MW 100 ablegen
     :BEA
FEHL :                Programmteil zur Fehlerbehandlung
     :
     :BE
```

Hinweis
Wenn Sie wie im Beispiel 2 das DBA-Register verstellen, wird das DBL- Register **nicht** verändert. Damit ist eine Transferfehler-überwachung nicht mehr gewährleistet!
Durch Anwendung des Sonderfunktions-OBs 180 "Variabler Datenbausteinzugriff" können Sie das DBA-Register ebenfalls um eine vorgegebene Anzahl an Datenwörtern verschieben. Da der OB 180 gleichzeitig das DBL-Register verändert, werden Transferfehler weiterhin überwacht.

Register 8:
DBL = Datenbaustein-Länge

Zusätzlich zum DBA-Register wird bei jedem Aufruf eines Datenbausteins das DBL-Register geladen. Es enthält die Länge (in Wörtern) des aufgerufenen Datenbausteins **ohne** Baustein-Kopf. Das DBL-Register wird vor jedem Aufruf des OB 1 oder FB 0 gleich ’0’ gesetzt.

Das DBL-Register bleibt **erhalten**, wenn

- durch eine Sprunganweisung (SPA/SPB) die Programmbearbeitung in einem anderen Baustein fortgesetzt wird

  oder

- eine andere Programmbearbeitungsebene eingeschachtelt wird.

Es **ändert** sich, wenn

- ein anderer Datenbaustein aufgeschlagen wird

  oder

- ein Rücksprung in einen übergeordneten Baustein erfolgt, nachdem im aufgerufenen Baustein ein neuer Datenbaustein aufgeschlagen wurde (siehe auch Abschnitt 2.4.2).

Beispiel

Wirkung der Operation **"AX DX 17"** auf das DBL:

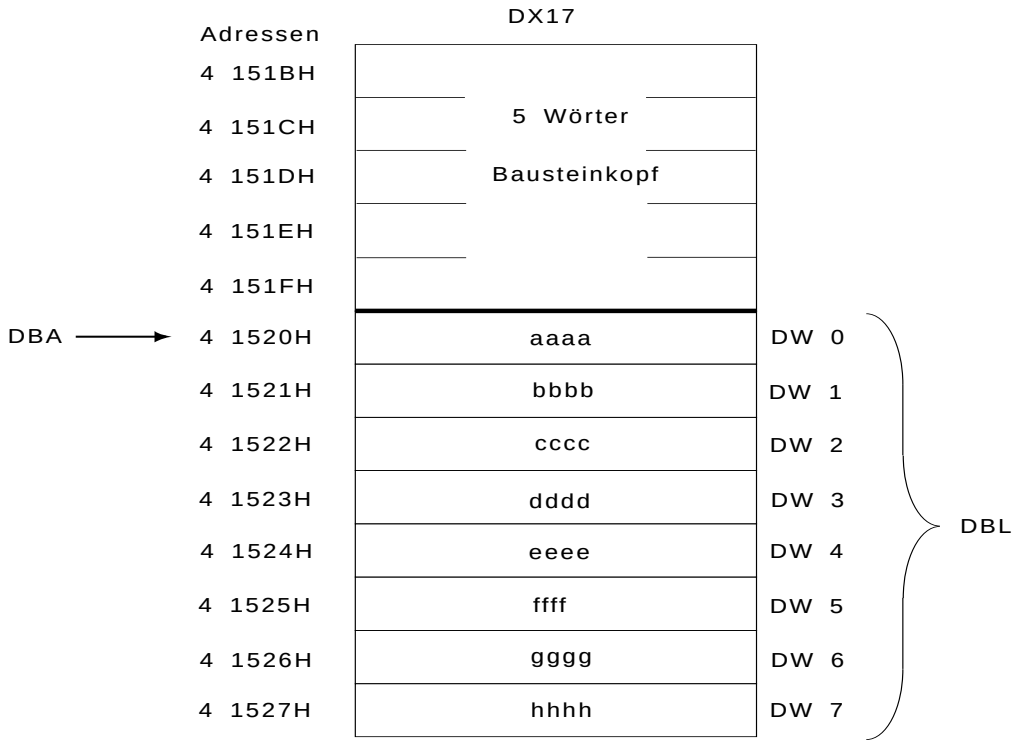


Bild 9-6 Verwendung des DBL-Registers

Bei Aufruf des DX 17 wird im DBL-Register die Anzahl der vorhandenen Datenwörter eingetragen, im Beispiel: DBL = **8** (DW 0 bis DW 7)

Hinweis: Im USTACK ist die im DBL-Register eingetragene Anzahl unter "DBL-REG" angegeben.

Register 15:
SAZ = STEP-Adreßzähler

Im Register 15 ist während der STEP-5-Programmbearbeitung die Absolutadresse derjenigen Operation im Programmspeicher eingetragen, die als nächste zu bearbeiten ist.

9.2.2 Beispiele für die Anwendung der Register

Beispiel 1: Alle Datenwörter eines Datenbausteins sollen mit einer Konstanten beschrieben werden.

Das unten dargestellte Programm beschreibt alle Datenwörter des DB 50 mit der Konstanten KH = A5A5. Nach Ändern der fettgedruckten STEP-5-Befehle kann es auch zum Beschreiben anderer Datenbausteine (DB oder DX) mit beliebigen Werten benutzt werden. Nicht vorhandene Datenbausteine oder Datenbausteine mit null Datenwörtern werden erkannt und führen zum Sprung zur Marke NIVO.
Die Anfangsadresse (DBA) und Länge (DBL) des Datenbausteins wird über die Sonderfunktion OB 181 "Datenbaustein (DB/DX) testen" ermittelt.

Das Programm nutzt alle vier Akkumulatoren. Im Bild sehen Sie die Belegung der Akkumulatoren während des Programmablaufs bis zur Marke SCHL. Innerhalb der Schleife ändert sich die Akkumulatorbelegung nicht.
Der AKKU 1 enthält zunächst die Adresse des letzten Datenwortes (DBA + DBL - 1) und wird mit jedem Schleifendurchlauf um eins vermindert.
Der AKKU 2 enthält die Adresse des ersten Datenwortes (DBA). Die Schleife wird abgebrochen, sobald der Inhalt des AKKU 1 kleiner als der Inhalt des AKKU 2 ist.

Zum Beschreiben der Datenwörter wird der Befehl TIR 10 verwendet, der den Inhalt des AKKU-3-L (die Konstante) unter der im AKKU-1-L stehenden Adresse abspeichert.

```
      :
      :L    KHA5A5     Konstante, mit der alle Datenwoerter
      :                   beschrieben werden sollen
      :L    KY 1,50    Typ und Nummer des Datenbausteins
      :ENT
      :SPA  OB 181     Sonderfunktions-OB "Datenbausteine testen"
      :SPB  =NIVO      Abbruch, falls DB 50 nicht vorhanden
      :TAK
      :ENT
      :+F
      :                AKKU 1 := Adresse des letzten Datenworts + 1
      :                AKKU 2 := Adresse des ersten Datenworts
      :                AKKU 3 := Konstante
      :!=F             Abbruch, falls der DB 50 null Datenwörter
      :SPB  =NIVO         enthaelt
      :
SCHL  :ADD  BF-1       Alle Datenwoerter, beginnend mit dem letzten
      :TIR  10            Datenwort, mit der im AKKU-3-L enthaltenen
      :                   Konstanten beschreiben
      :><F             Abfrage, ob 1. Datenwort erreicht
      :
      :SPB  =SCHL      Ruecksprung in die Schleife, wenn 1. Datenwort
      :                  noch nicht erreicht
      :                Fortsetzung des Programms...
```

Fortsetzung auf der nächsten Seite

Fortsetzung von Beispiel 1:

```
WEIT  :                nachdem alle Datenwörter beschrieben
   .  :                   wurden
      :BEA
NIVO  :  .             falls der DB 50 nicht vorhanden ist
      :                   oder null Datenwörter enthält.
      :BE
```

Hinweis: Der Programmteil ab Marke "SCHL" kann zum Beschreiben beliebiger Speicherbereiche (z. B. Merker, Zeiten, Zähler) mit einer Konstanten benutzt werden.

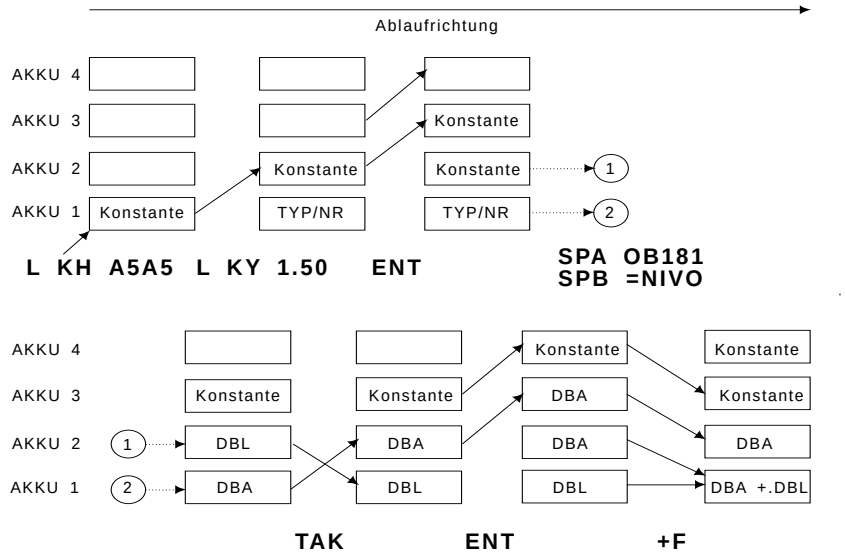


Bild 9-7 Belegung der AKKUs während des Programmablaufs

9

Beispiel 2: Löschen aller Merkerbytes (MB 0 bis MB 255)

```
      :L    KB 0          Konstante, mit der alle Merkerbytes beschrieben
      :                      werden sollen
      :L    BS 14         Basisadresse des Merkerbereichs (= Adressse des
      :                      ersten Merkerbytes MB 0)
      :ENT
      :L    KF +256       + Laenge des Merkerbereichs
      :ENT                   = (Adresse des letzten Merkerbytes MB 255) + 1
   :  +F
      :
SCHL  :ADD  BF -1         Alle 256 Merkerbytes, beginnend mit dem Merker-
      :TIR  10               byte MB255, mit der im AKKU-3-LL enthaltenen
      :                      Konstanten beschreiben
      :SPB  =SCHL
      :
      :
      :
```

9.3 Speicherblöcke transferieren

Anwendung

Mit den Systemoperationen TNB und TNW können Sie Speicherblöcke transferieren (max. 255 byte mit TNB, max. 255 Wörter mit TNW). Sie können mit diesen Operationen sowohl auf den lokalen Speicherbereich als auch auf den byteweise organisierten Teil des globalen Speicherbereichs (Adressen F000H bis F3FFH, FC00H bis FFFFH) zugreifen.

Operationen

Tabelle 9-3 Operationen für Blocktransfer

Operation	Operand	Funktion
TNB	0 bis 255	Blocktransfer 0 bis 255 byte
TNW	0 bis 255	Blocktransfer 0 bis 255 Wörter

Parameter

Blocklänge

Operand = Anzahl Bytes (TNB) bzw. Anzahl Wörter (TNW)

Endadresse des Quellbereichs

AKKU-2-L = Endadresse des Quellbereichs

Endadresse des Zielbereichs

AKKU-1-L = Endadresse des Zielbereichs

Quellbereich und Zielbereich müssen **vollständig** in einem der in Tabelle 9-4 genannten Speicherbereiche liegen und **dürfen sich nicht überschneiden**.

erlaubte Speicherbereiche

Tabelle 9-4 Für TNB und TNW erlaubte Speicherbereiche

Adressen	Speicherbereich
	Anwenderspeicher:
0000H bis 1FFFH	Anwender-Modul (16 bit) 8K-Wörter
0000H bis 3FFFH	Anwender-Modul (16 bit) 16K-Wörter
0000H bis 7FFFH	Anwender-Modul (16 bit) 32K-Wörter

Adressen	Speicherbereich
Fortsetzung der Tabelle 9-4:	
	System-RAM:
8000H bis DD7FH	DB-RAM (16 bit)
DD80H bis E3FFH	DB 0 (16 bit)
E400H bis E7FFH	S-Merker (8 bit)
E800H bis EDFFH	Systemdaten (16 bit: BA, BB, BS, BT, Zähler und Zeiten)
EE00H bis EFFFH	RAM (8 bit: Merker, Prozeßabbild)
F000H bis FFFFH	Peripherie (8 bit)/S5-Bus

Ablauf

Der Blocktransfer erfolgt dekrementierend, d. h. er beginnt seine Übertragung mit der höchsten Adresse des Quellbereichs (=Endadresse) und beendet sie mit der niedrigsten.

Anwendung auf den Kachelbereich

Die Operationen TNB und TNW sind im Mehrprozessor-Automatisierungsgerät AG S5-135U **nicht** für den Zugriff auf den Kachelbereich (Adressen F400H - FBFFH) geeignet. Verwenden Sie stattdessen die Operationen aus Abschnitt 9.4.4 "Zugriff auf den Kachelspeicher" oder die Sonderfunktionen aus Kapitel 6.21 "Kachelzugriffe".

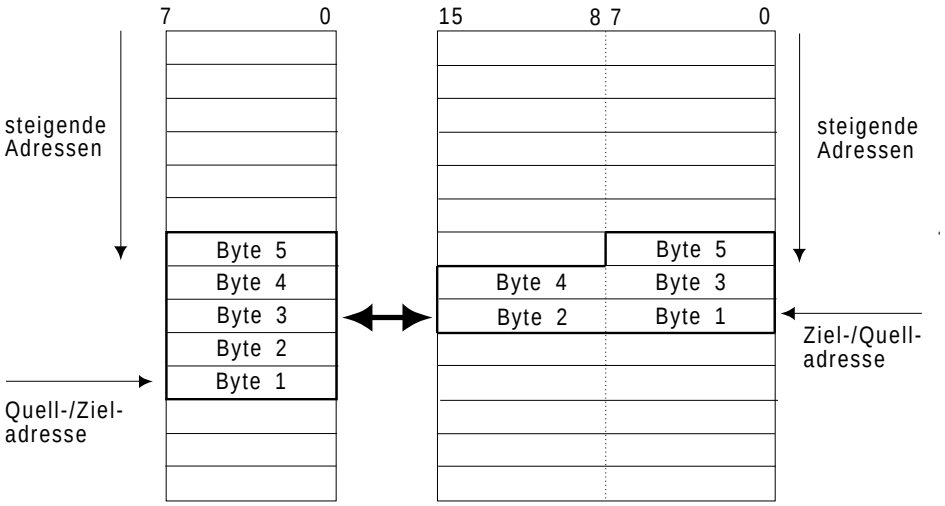
Besonderheiten

Pseudobefehlsgrenzen bei TNB und TNW

Die Operationen TNB und TNW sind langlaufende STEP-5-Operationen, die sogenannte "Pseudobefehlsgrenzen" enthalten: D .h.: Die Datenübertragung erfolgt je nach Quell- und Zielbereich in Teilblöcken unterschiedlicher Größe. Wenn nun während der Übertragung eines Teilblocks ein Fehler (z. B. Zyklusfehler) oder eine Unterbrechung (z. B. durch Weck- oder Prozeßalarm) auftritt, so wird am Ende dieses Teilblocks an der Pseudobefehlsgrenze der entsprechende Organisationsbaustein eingeschachtelt. Voraussetzung für den Aufruf des Prozeßalarm-OBs oder eines Weckalarm-OBs an einer Pseudobefehlsgrenze ist, daß im DX 0 "Unterbrechbarkeit an Befehlsgrenzen" eingestellt ist.

Tritt während der Übertragung einmal oder mehrmals Quittungsverzug und/oder Adressierfehler auf, so werden zuerst alle Teilblöcke übertragen und dann vor der Ausführung der nächsten Operation einmalig der dafür vorgesehene Fehler-Organisationsbaustein aufgerufen (bei QVZ und ADF gleichzeitig nur der QVZ-OB). Als Fehleradresse wird immer **die** Adresse angegeben, an der **zuerst** ein Fehler aufgetreten ist. Da TNB und TNW dekrementierend arbeiten, ist dies bei **mehrfachen** Fehleradressen die **höchste** Fehleradresse in dem Bereich, in dem **zuerst** ein Fehler aufgetreten ist.

TNB und TNW zwischen 8- und 16-bit-Speicherbereichen



Übertragen der Bytes 1 bis 5:

```
:L <Quelladresse>
:L <Zieladresse>
:TNB 5
```

Übertragen der Bytes 1 bis 4:

```
:L <Quelladresse>
:L <Zieladresse>
:TNW 2
```

Bild 9-8 Transferieren von Speicherblöcken

9.3.1
Beispiel für Übertragen von Speicherblöcken

a) Aufgabenstellung:

Kopieren eines Blocks von max. 4095 Datenwörtern aus einem DB- oder DX-Datenbaustein in einen anderen DB- oder DX-Datenbaustein. Der Anfang des Blocks wird innerhalb des Quell- und Ziel-Datenbausteins durch je einen Offset-Wert zwischen 0 und 4095 festgelegt.

Das Programm wird im Baustein FB 10 hinterlegt.

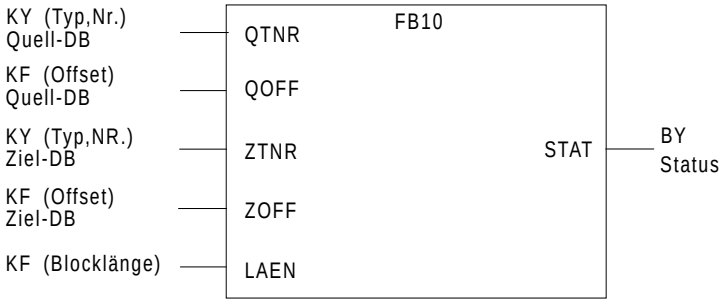
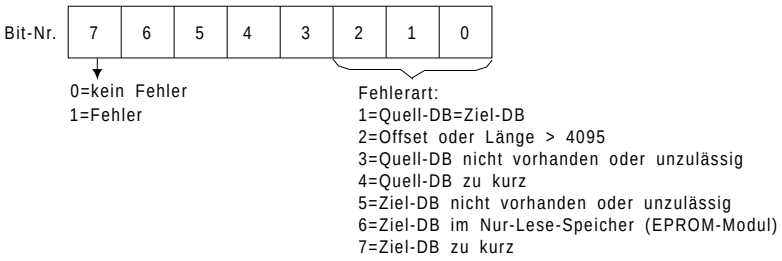


Bild 9-9 Funktionsbaustein für Datenblockverkehr

Vor dem Kopieren werden die Eingangsparameter geprüft. Im Fehlerfall wird im Ausgangsparameter STAT das Bit Nr. 7 = 1 gesetzt und in den Bits Nr. 0 bis Nr. 2 die Fehlerart angegeben:



Fortsetzung auf der nächsten Seite

Fortsetzung 1 des Beispiels:

b) Programmstruktur:

Der FB 10 gliedert sich in fünf Programmabschnitte mit folgenden Aufgaben:

- Eingangsparameter:

  a) Prüfen, ob Quell- und Ziel-Datenbaustein nicht denselben Typ und dieselbe Nummer haben.
  b) Prüfen, ob die Eingangsparameter "Quell-Offset", "Ziel-Offset" und "Blocklänge" kleiner 4096 sind.

- Quell-Datenbaustein:

  a) Prüfen, ob der Quell-Datenbaustein vorhanden und lang genug ist.
  b) Berechnen der absoluten Adresse des letzten Datenworts im Ziel-Block.

- Ziel-Datenbaustein:

  a) Prüfen, ob der Ziel-Datenbaustein vorhanden und lang genug ist und ob er im Schreib-Lese-Speicher (RAM-Modul oder DB-RAM) liegt.
  b) Berechnen der absoluten Adresse des letzten Datenworts im Ziel-Block.

- Transfer:

  Kopiervorgang durchführen mit Hilfe des TNW-Befehls.
  Blöcke mit mehr als 255 Wörtern werden in Teilblöcken zu je 128 Wörtern übertragen (Befehl TNW 128).
  Ein eventuell noch vorhandener Rest wird durch einen zusätzlichen TNW-Befehl übertragen.

- Anzeige:

  Versorgen des Ausgangsparameters "Status" entsprechend dem Ergebnis der vorgenommenen Prüfungen.

c) Belegte Speicherzellen:

```
MW 242  Endadresse des Datenziels
MW 244  Endadresse der Datenquelle
MW 246  Blocklänge

MW 248  Offset im Ziel-Datenbaustein
MW 250  Typ und Nummer des Ziel-Datenbausteins

MW 252  Offset im Quelldatenbaustein
MW 254  Typ und Nummer des Quell-Datenbausteins

BS 60   Teil-Block-Zähler
```

Fortsetzung auf der nächsten Seite

Fortsetzung 2 des Beispiels:

b) Programmierung des Funktinsbausteins FB 10:

Hinweis: Soll ab Datenwort DW 0 kopiert werden, entfallen die fett-kursiv gedruckten Programmteile. Es wird kein Offset-Wert angegeben.

```
FB10

NETZWERK 1
NAME:DB-DB-TR                      DATENBAUST.-DATENBAUST.-TRANSFER
BEZ   :QTNR E/A/D/B/T/Z:     D     KM/KH/KY/KC/KF/KT/KZ/KG:    KY
BEZ   :QOFF E/A/D/B/T/Z:     D     KM/KH/KY/KC/KF/KT/KZ/KG:KF
BEZ   :ZTNR E/A/D/B/T/Z:     D     KM/KH/KY/KC/KF/KT/KZ/KG:    KY
BEZ   :ZOFF E/A/D/B/T/Z:     D     KM/KH/KY/KC/KF/KT/KZ/KG:KF
BEZ   :LAEN E/A/D/B/T/Z:     D     KM/KH/KY/KC/KF/KT/KZ/KG:    KF
BEZ   :STAT   E/A/D/B/T/Z:   A     BI/BY/W/D: BY
      :
      :                   BEGINN EINGANGS-PARAMETER
      :LW   =QTNR         TYP (DB/DX) UND NUMMER DES
      :T    MW 254           QUELL-DATENBAUSTEINS
      :LW   =ZTNR         TYP (DB/DX) UND NUMMER DES
      :T    MW 250           ZIEL-DATENBAUSTEINS
      :!=F                QUELL-DB = ZIEL-DB ?
      :SPB  =F001         SPRUNG, FALLS JA
      :
      :
      :
      :LW=  QOFF          OFFSET IM QUELL-
      :T    MW 252           DATENBAUSTEIN
      :LW=  ZOFF          OFFSET IM ZIEL-
      :T    M  W 248          DATENBAUSTEIN
      :OW
      :LW   =LAEN         LAENGE (ANZAHL DATENWOERTER) DES
      :T MW 246               ZU TRANSFERIERENDEN BLOCKES
      :                       (BLOCK-LAENGE)
      :OW                 ODER (QUELL-OFFSET/ZIEL-OFFSET)
      :L KH F000             LAENGE >= 4096 ?
      :UW                 SPRUNG, FALLS JA
      :SPP  =F002         END EINGANGS-PARAMETER
      :
      :
      :
      :
      :
      :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 3 des Beispiels:

```
                          BEGINN QUELL-DATENBAUSTEIN
     :L    MW 254         TYP U. NUMMER DES QUELL-DATENBAUSTEINS
     :SPA  OB 181         DATENBAUSTEIN TESTEN
     :SPB  =F003          SPRUNG, FALLS BST.-TEST NEGATIV
     :TAK                 A1: ANZAHL DW , A2: ADRESSE
     :ENT                 A3: ADRESSE
     :L    MW 252         OFFSET IM QUELL-DATENBAUSTEIN
     :ENT                 A3: ANZAHL DW, A4: ADRESSE
     :L    MW 246         BLOCK-LAENGE
     :+F                  OFFSET + BLOCK-LAENGE
     :<F                  ANZ. DW < OFFSET + BL.-LAENGE ?
     :SPB  =F004          SPRUNG, FALLS JA
     :L    KB 1           A2: OFFSET + BL.-L., A3: ADRESSE
     :-F                  OFFSET + BLOCK-LAENGE - 1
     :+F                  OFFSET + BL.-L. - 1 + ADRESSE
     :T    MW 244         END-ADRESSE DER DATEN-QUELLE
     :                    END QUELL-DATENBAUSTEIN
     :
     :
     :
     :
     :
     :                    BEGINN ZIEL-DATENBAUSTEIN
     :L    MW 250         TYP U. NUMMER DES ZIEL-DATENBAUSTEINS
     :SPA  OB 181         DATENBAUSTEIN TESTEN
     :SPB  =F005          SPRUNG, FALLS BST.-TEST NEGATIV
     :SPM  =F006          SPRUNG, FALLS BST. IM EPROM
     :TAK                 A1: ANZAHL DW, A2: ADRESSE
     :ENT                 A3: ADRESSE
     :L    MW 248         OFFSET IM ZIEL-DATENBAUSTEIN
     :ENT                 A3: ANZAHL DW, A4: ADRESSE
     :L    MW 246         BLOCK-LAENGE
     :+F                  OFFSET + BLOCK-LAENGE
     :<F                  ANZ. DW < OFFSET + BL.-LAENGE ?
     :SPB  =F007          SPRUNG, FALLS JA
     :L    KB 1           A2: OFFSET + BL.-L., A3: ADRESSE
     :-F                  OFFSET + BLOCK-LAENGE - 1
     :+F                  OFFSET + BL.-L. - 1 + ADRESSE
     :T    MW 242         END-ADRESSE DES DATEN-ZIELS
     :                    END ZIEL-DATENBAUSTEIN
     :
     :
     :
     :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 4 des Beispiels:

```
      :                    BEGINN TRANSFER
      :L    KB 0           VERGLEICHSWERT
      :L    MB 246         BLOCK-LAENGE, HIGH-BYTE
      :!=F                 BLOCK-LAENGE >= 256 WORTE ?
      :SLW  1              MULTIPL. MIT 2, ANZAHL TEIL-
      :T    BS 60           BLOECKE MIT JE 128 WORTEN
      :L    MW 244         END-ADRESSE DER DATEN-QUELLE
      :L    MW 242         END-ADRESSE DES DATEN-ZIELS
      :SPB  =REST          SPRUNG, FALLS BLOCK-L. <256 W.
SCHL  :TNW  128            TRANSFER EINES TEIL-BLOCKS
      :ADD  KF -128        QUELL-END-ADRESSE UM LAENGE DES
      :TAK                    TEIL-BLOCKS REDUZIEREN
      :ADD  KF -128        ZIEL-END-ADRESSE UM LAENGE DES
      :TAK                    TEIL-BLOCKS REDUZIEREN
      :SPA  OB 160         ZAEHL-SCHLEIFE
      :SPB  =SCHL          SPRUNG, FALLS NICHT ALLE
      :                       TEIL-BLOECKE TRANSFERIERT
REST  :B    MW 246         BLOCK-LAENGE, LOW-BYTE
      :TNW  0              REST-BLOCK TRANSFERIEREN
      :                    END TRANSFER
      :
      :
      :
      :
      :
      :                    BEGINN ANZEIGE
      :L    KB 0           KENNUNG 00 (HEX.): KEIN FEHLER
ENDE  :T    =STAT          AUSGANGS-PARAMETER STATUS/FEHLER
      :BEA
F001  :L    KB 129         FEHLER-KENNUNG 81 (HEX.):
      :SPA  =ENDE             QUELL-DB = ZIEL-DB
F002  :L    KB 130         FEHLER-KENNUNG 82 (HEX.):
      :SPA  =ENDE             OFFSET ODER   LAENGE >= 4096
F003  :L    KB 131         FEHLER-KENNUNG 83 (HEX.):
      :SPA  =ENDE             QUELL-DB UNZULAESSIG
F004  :L    KB 132         FEHLER-KENNUNG 84 (HEX.):
      :SPA  =ENDE             QUELL-DB ZU KURZ
F005  :L    KB 133         FEHLER-KENNUNG 85 (HEX.):
      :SPA  =ENDE             ZIEL-DB UNZULAESSIG
F006  :L    KB 134         FEHLER-KENNUNG 86 (HEX.):
      :SPA  =ENDE             ZIEL-DB IM NUR-LESE-SPEICHER
F007  :L    KB 135         FEHLER-KENNUNG 87 (HEX.):
      :SPA  =ENDE             ZIEL-DB ZU KURZ
      :                    END ANZEIGE
      :BE
```

9.4 Operationen mit dem Basisadressregister (BR-Register)

Anwendung

Das Basisadreßregister (32 bit) ermöglicht Ihnen Adreßrechnungen und indirekte Register-Lade- und -Transferoperationen ohne Verwendung der AKKUs für die Adressierung.
Zugegriffen wird auf die Speicherzelle, deren absolute Adresse sich als Summe von BR-Registerinhalt und einer Konstanten errechnet:

Absolute Adresse = BR-Registerinhalt + Konstante

Operationen

Tabelle 9-5 Lade- und Rechenoperationen mit dem BR-Register

Operation	Operand	Funktion
MBR	Konstante (0H bis F FFFFH)	das BR-Register mit einer 20-bit-Konstanten laden [1]
ABR	Konstante (-32 768 bis +32 767)	eine 16-bit-Konstante zum Inhalt des BR-Registers addieren

[1] Die Bits 2^{20} bis 2^{31} des BR-Registers werden gleich '0' gesetzt.

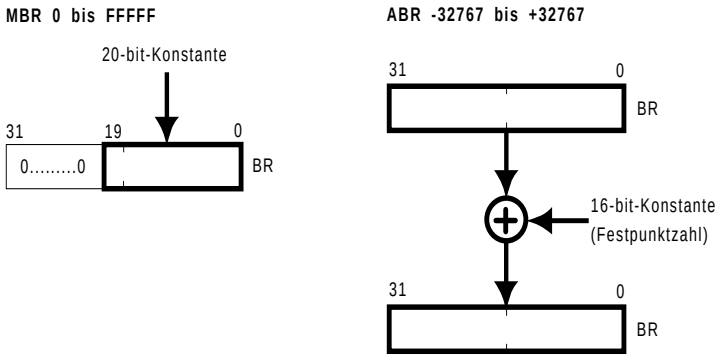


Bild 9-10 Laden des BR-Registers

Veränderung des BR-Registers

- Das BR-Register bleibt **erhalten**, wenn durch eine Sprunganweisung ('SPA FB'/'SPB FB') in einem **anderen Baustein derselben Programmbearbeitungsebene** fortgesetzt wird.
- Das BR-Register bleibt nach **Einschachtelung** einer anderen Programmbearbeitungsebene **erhalten**.
- Bei **Aufruf** einer anderen Programmbearbeitungsebene durch das Systemprogramm wird das BR-Register auf **'0'** gesetzt

9.4.1 Transferoperationen zwischen Registern

Anwendung

Die in diesem Abschnitt aufgeführten Operationen können Sie dazu benutzen, schnell Werte zwischen den Registern AKKU 1, STEP-Adreßzähler (SAZ) und Basisadreßregister (BR) auszutauschen.

Operationen

Tabelle 9-6 Register-Register-Operationen

Operation	Operand	Funktion
MAS	–	den Inhalt des AKKU 1 (Bit 2^0 bis 2^{14}) in den STEP-Adreßzähler transferieren
MAB	–	den Inhalt des AKKU 1 (Bit 2^0 bis 2^{31}) in das Basisadreßregister transferieren
MSA	–	den Inhalt des STEP-Adreßzählers in den AKKU 1 transferieren [1)]
MSB	–	den Inhalt des STEP-Adreßzählers in das Basisadreßregister transferieren [1)]
MBA	–	den Inhalt des Basisadreßregisters in den AKKU 1 transferieren
MBS	–	den Inhalt des Basisadreßregisters (Bit 2^0 bis 2^{14}) in den STEP-Adeßzähler transferieren

[1)] Die Bits 2^{15} bis 2^{31} werden auf '0' gesetzt.

Wie die Register bei den Operationen verändert werden, zeigt Ihnen das folgende Bild.

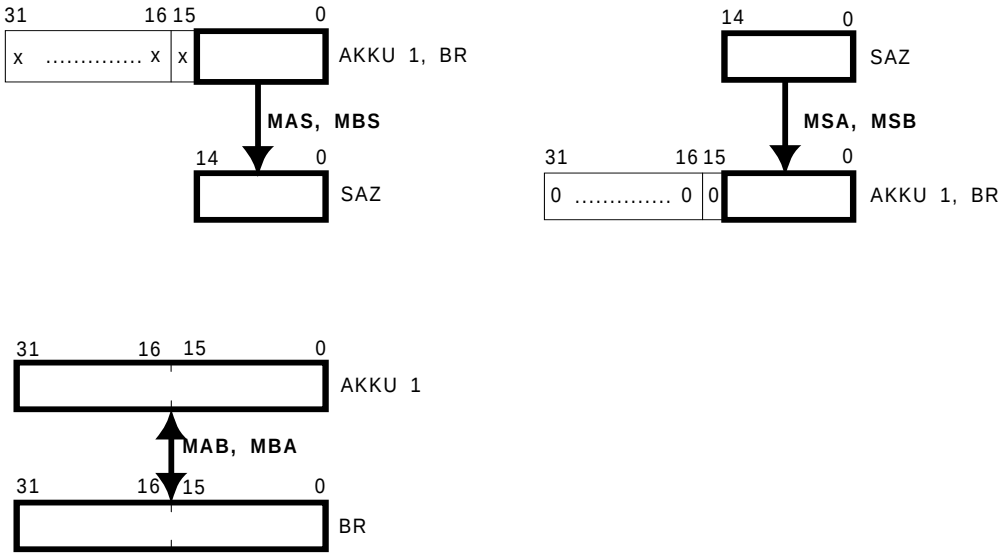


Bild 9-11 Register-Register-Transferoperationen

9.4.2 Zugriffe auf den lokalen Speicher

Anwendung

Die folgenden Operationen ermöglichen den Zugriff auf den lokalen, wortweise organisierten Speicher über eine absolute Speicheradresse. Die Absolutadresse ist die Summe vom BR-Registerinhalt und der im Befehl enthaltenen 16-bit-Konstanten (-32768 bis +32767).

Operationen

Tabelle 9-7 Operationen für Zugriffe auf den lokalen Speicher

Operation	Operand	Beschreibung
LRW	Konstante (-32768 bis +32767)	die angegebene Konstante zum Inhalt des BR-Registers addieren [1] und das so adressierte Wort in den AKKU-1-L laden
LRD	Konstante (-32768 bis +32767)	die angegebene Konstante zum Inhalt des BR-Registers addieren [1] und das so adressierte Doppelwort in den AKKU 1 laden
TRW	Konstante (-32768 bis +32767)	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-L in das so adressierte Wort transferieren

Operation	Operand	Beschreibung
Fortsetzung der Tabelle 9-7:		
TRD	Konstante (-32768 bis +32767)	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU 1 in das so adressierte Doppelwort transferieren

[1] AKKU 2 $_{neu}$ = AKKU 1$_{alt}$

zulässiger Adreßbereich

Die Abolutadresse muß liegen

- bei LRW, TRW: zwischen 0000H und E3FFH oder E800H und EDFFH,
- bei LRD, TRD: zwischen 0000H und E3FEH oder E800H und EDFEH.

Fehlerreaktion

Liegt die errechnete Adresse der Speicherzelle nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

9.4.3 Zugriffe auf den globalen Speicher

Anwendung

Mit den folgenden Operationen können Sie auf den globalen, **byte- oder wortweise** organisierten Speicher über eine absolute Speicheradresse zugreifen. Die Absolutadresse ist die Summe vom BR-Register-Inhalt und der im Befehl enthaltenen Konstanten (-32768 bis 32767).

Testen und Setzen einer Belegtzelle im Globalbereich

Der Zugriff einzelner CPUs auf gemeinsam genutzte Speicherbereiche kann über eine Belegtzelle gesteuert werden. Jedem gemeinsam genutzten Speicherbereich wird eine Belegtzelle zugeordnet, die von allen beteiligten CPUs vor jedem Zugriff getestet werden muß. Die Belegtzelle enthält entweder den Wert ’0’ oder die Steckplatzkennung der CPU, die gerade den Speicherbereich benutzt und ihn durch **Beschreiben der Belegtzelle mit ’0’ wieder freigeben muß**. (Beachten Sie auch die Erläuterungen zu den Operationen "Semaphor setzen/SES" und "Semaphor freigeben/SEF" in Abschnitt 3.5.5.).

Die Operation TSG bearbeitet das Testen und Setzen einer Belegtzelle.

Operation	Operand	Beschreibung
TSG	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und die so adressierte Belegtzelle testen und setzen

Ablauf

Als Belegtzelle wird das Low-Byte des durch BR-Register + Konstante adressierten Wortes verwendet. Falls der Inhalt des Low-Bytes '0' ist, trägt der TSG-Befehl die Steckplatzkennung (aus BS 29) in die Belegtzelle ein.

Das Testen (= Lesen) und das eventuelle Belegen (= Schreiben) bilden eine Programmeinheit, die nicht unterbrochen werden kann.

Ergebnis

Das Testergebnis ist über die Anzeigen ANZ 0 und ANZ 1 auswertbar:

ANZ 1	ANZ 0	Bedeutung
0	0	Inhalt der Belegtzelle ist '0'; die CPU trägt ihre Steckplatzkennung ein.
1	0	Die eigene Steckplatzkennung ist bereits in der Belegtzelle eingetragen.
0	1	Die Belegtzelle enthält eine fremde Steckplatzkennung.

Hinweis
Die Operation TSG muß von **allen** CPUs verwendet werden, die synchronisiert auf einen **gemeinsamen globalen Speicherbereich** zugreifen sollen.

zulässiger Adreßbereich

Die Absolutadresse muß zwischen 0000H und EFFFH liegen.

Fehlerreaktion

Liegt die errechnete Adresse der Speicherzelle nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Lade und Tranfer-operationen auf den byte-weise organisierten globalen Speicher

Tabelle 9-8 Operationen für Zugriffe auf den byteweise organisierten globalen Speicher

Operation	Operand	Beschreibung
LB GB	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Byte in den AKKU-1-LL laden [1) 3)]
LB GW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Wort in den AKKU-1-L laden [2) 3)]
LB GD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Doppelwort in den AKKU 1 laden [3)]
TB GB	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-LL in das so adressierte Byte transferieren
TB GW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-L in das so adressierte Wort transferieren
TB GD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU 1 in das so adressierte Doppelwort transferieren

[1)] AKKU-1-LH und AKKU-1-H werden auf ’0’ gesetzt.

[2)] AKKU-1-H wird auf ’0’ gesetzt.

[3)] AKKU 2 $_{neu}$: = AKKU 1$_{alt}$

zulässiger Adreßbereich

Die Absolutadresse muß liegen

- bei LB GB, TB GB: zwischen 0000H und EFFFH ,
- bei LB GW, TB GW: zwischen 0000H und EFEH,
- bei LB GD, TB GD: zwischen 0000H und EFFCH.

Fehlerreaktion

Liegt die errechnete Adresse der Speicherzelle nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Lade- und Transfer-operationen für den wort-weise organisierten globalen Speicher

Tabelle 9-9 Operationen für Zugriffe auf den wortweise organisierten globalen Speicher

Operation	Operand	Beschreibung
LW GW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Wort in den AKKU-1-L laden [1) 2)]
LW GD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Doppelwort in den AKKU 1 laden [2)]
TW GW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-L in das so adressierte Wort transferieren
TW GD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU 1 in das so adressierte Doppelwort transferieren

[1)] AKKU-1-H wird auf ’0’ gesetzt.

[2)] AKKU 2_{neu} : = AKKU 1_{alt}

zulässiger Adreßbereich

Die Absolutadresse muß liegen

- bei LW GW, TW GW: zwischen 0000H und EFFFH,
- bei LW GD, TW GD zwischen 0000H und EFFEH.

Fehlerrreaktion

Liegt die errechnete Adresse der Speicherzelle nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

9.4.4 Zugriffe auf den Kachelspeicher

Anwendung

Mit den folgenden Operationen können Sie auf **byte- oder wortweise** organisierte Kacheln über eine absolute Speicheradresse zugreifen. Die Absolutadresse ist die Summe vom BR-Register-Inhalt und der im Befehl enthaltenen Konstanten (-32768 bis 32767).

Prinzipieller Ablauf des Kachelzugriffs

Der Globalbereich enthält im Bereich der Adressen F400H bis FBFFH ein "Fenster" zum Einblenden eines von max. 256 Speicherbereichen (= Kacheln). Eine Kachel belegt max. 2K Adressen und kann byteweise oder wortweise organisiert sein. Vor jedem Zugriff auf den Kachelbereich wird eine der 256 Kacheln durch Eintrag ihrer Kachelnummer in das **Select-Register** (Kachel-Adreßregister) ausgewählt. Der Vorgang "Beschreiben des Select-Registers und anschließender Zugriff auf den Kachelbereich" ist ununterbrechbar.

Vor jedem Zugriff (Laden/Transferieren) auf den Kachelbereich muß eine der 256 Kacheln aufgeschlagen werden. Dazu übergeben Sie die Nummer der aufzuschlagenden Kachel im AKKU-1-L; diese Nummer wird durch den Befehl ACR in das CPU-interne **Kachel-Register** eingetragen. Alle folgenden Kacheloperationen schreiben vor dem Kachelzugriff den Inhalt des Kachel-Registers in das Select-Register der entsprechenden Baugruppen auf dem S5-Bus.

Veränderung des Kachel-Registers

- Das Kachel-Register **bleibt erhalten**, wenn ein anderer Baustein derselben Programmbearbeitungsebene aufgerufen wird.
- Wenn das Kachelregister in einem Baustein verändert wird, so bleibt sein **Wert erhalten**, wenn am Bausteinende in den aufrufenden Baustein zurückgesprungen wird.
- Das Kachelregister wird **nach** Einschachtelung einer anderen Programmbearbeitungsebene durch das Systemprogramm mit **demselben Wert** geladen, den es **vor** der Einschachtelung hatte.
- Bei Aufruf einer **anderen Programmbearbeitungsebene** durch das Systemprogramm wird das Kachelregister **'0'** gesetzt.

Aufschlagen einer Kachel

Operation	Parameter	Beschreibung
ACR		diejenige Kachel aufschlagen, deren Nummer im AKKU-1-L steht, zulässige Werte: 0 bis 255

Fehlerreaktion

Die Kachelnummer muß zwischen 0 und 255 liegen. Ist dies nicht der Fall, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Testen und Setzen einer Belegtzelle im Kachelbereich

Der Zugriff einzelner CPUs auf gemeinsam genutzte Speicherbereiche kann über eine Belegtzelle gesteuert werden. Jedem gemeinsam genutzten Speicherbereich wird eine Belegtzelle zugeordnet, die von allen beteiligten CPUs vor jedem Zugriff getestet werden muß. Die Belegtzelle enthält entweder den Wert ’0’ oder die Steckplatzkennung der CPU, die gerade den Speicherbereich benutzt und ihn durch **Beschreiben der Belegtzelle mit ’0’ wieder freigeben muß**. (Beachten Sie auch die Erläuterungen zu den Operationen "Semaphor setzen/SES" und "Semaphor freigeben/SEF" in Abschnitt 3.5.5.)

Der Befehl TSC bearbeitet das Testen und Setzen einer Belegtzelle auf der aufgeschlagenen Kachel.

Operation	Operand	Beschreibung
TSC	-32 768 bis +32 767	die angegebene Konstante zum Inhalt des BR-Registers addieren und die so adressierte Belegtzelle auf der aufgeschlagenen Kachel testen und setzen

Ablauf

Als Belegtzelle wird das Low-Byte des durch BR-Register + Konstante adressierten Wortes verwendet. Falls der Inhalt des Low- Bytes ’0’ ist, trägt der TSC-Befehl die Steckplatzkennung in die Belegtzelle ein.

Das Testen (= Lesen) und das eventuelle Belegen (= Schreiben) bilden eine Programmeinheit, die nicht unterbrochen werden kann.

Ergebnis

Das Ergebnis der Operation TSC ist über die Anzeigen ANZ 0 und ANZ 1 auswertbar:

ANZ 1	ANZ 0	Bedeutung
0	0	Inhalt der Belegtzelle ist ’0’; die CPU trägt ihre Steckplatzkennung ein.
1	0	Die eigene Steckplatzkennung ist bereits in der Belegtzelle eingetragen.
0	1	Die Belegtzelle enthält eine fremde Steckplatzkennung.

Hinweis
Die Operation TSC muß von **allen** CPUs verwendet werden, die **synchronisiert** auf einen **gemeinsamen globalen Speicherbereich** (Kachelbereich) zugreifen sollen**.**

Fehlerreaktion

Die Belegtzelle muß auch auf der entsprechenden Baugruppe und in der gemeinsam benutzten Kachel zwischen F400H und FBFFH liegen. Ist dies nicht der Fall, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Lade- und Transferoperationen für byteweise organisierte Kacheln

Tabelle 9-10 Operationen für Zugriffe auf byteweise organisierte Kacheln

Operation	Operand	Beschreibung
LB CB	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Byte in der aufgeschlagenen Kachel in den AKKU-1-LL laden [1) 3)]
LB CW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Wort in der aufgeschlagenen Kachel in den AKKU-1-L laden [2) 3)]
LB CD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Doppelwort in der aufgeschlagenen Kachel in den AKKU 1 laden [3)]

Operation	Operand	Beschreibung
Fortsetzung der Tabelle 9-10:		
TB CB	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-LL in das so adressierte Byte in der aufgeschlagenen Kachel transferieren
TB CW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-L in das so adressierte Wort in der aufgeschlagenen Kachel transferieren
TB CD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU 1 in das so adressierte Doppelwort in der aufgeschlagenen Kachel transferieren

[1] AKKU-1-LH und AKKU-1-H werden auf ’0’ gesetzt.

[2] AKKU-1-H wird auf ’0’ gesetzt.

[3] AKKU 2 $_{neu}$: = AKKU 1$_{alt}$

zulässiger Adreßbereich

Die Absolutadresse muß liegen

- bei LB CB, TB CB: zwischen F400H und FBFFH,
- bei LB CW, TB CW: zwischen F400H und FBFEH,
- bei LB CD, TB CD: zwischen F400H und FBFCH.

Fehlerreaktion

Liegt die errechnete Byteadresse nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Lade- und Transfer-operationen für wortweise organisierte Kacheln

Tabelle 9-11 Operationen für Zugriffe auf wortweise organisierte Kacheln

Operation	Operand	Beschreibung
LW CW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Wort in der aufgeschlagenen Kachel in den AKKU-1-L laden [1]
LW CD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und das so adressierte Doppelwort in der aufgeschlagenen Kachel in den AKKU 1 laden [2]
TW CW	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU-1-L in das so adressierte Wort in der aufgeschlagenen Kachel transferieren
TW CD	-32768 bis +32767	die angegebene Konstante zum Inhalt des BR-Registers addieren und den Inhalt des AKKU 1 in das so adressierte Doppelwort in der aufgeschlagenen Kachel transferieren

[1] AKKU-1-H wird auf ’0’ gesetzt.

[2] AKKU 2_{neu}: = AKKU 1_{alt}

9

zulässiger Adreßbereich

Die Absolutadresse muß liegen

- bei LW CW, TW CW: zwischen F400H und FBFFH
- bei LW CD, TW CD: zwischen F400H und FBFEH.

Fehlerrreaktion

Liegt die errechnete Adresse der Speicherzelle nicht im zulässigen Adreßbereich, so erkennt die CPU einen Laufzeitfehler und ruft den **OB 31** auf, falls er geladen ist. Ist der OB 31 nicht geladen, so geht die CPU in den Stoppzustand.
In beiden Fällen sind im AKKU-1-L Fehlerkennungen hinterlegt, die den aufgetretenen Fehler näher erläutern (siehe Abschnitt 5.7.2).

Mehrprozessorbetrieb und Mehrprozessorkommunikation

10

Inhalt von Kapitel 10

10 Mehrprozessorbetrieb und Mehrprozessorkommunikation

Dieses Kapitel gibt Ihnen zunächst einige Hinweise, wann Sie den Mehrprozessorbetrieb anwenden können und wie der Datenaustausch der CPU erfolgt. Es gibt Ihnen Informationen darüber, was Sie als Programmierer für den Mehrprozessorbetrieb tun und beachten müssen (Abschnitt 10.1).
Schließlich erhalten Sie ausführliche Anleitungen mit Anwendungsbeispielen für den Austausch größerer Datenmengen im Mehrprozessorbetrieb (Mehrprozessorkommunikation – Abschnitte - 10.2 bis 10.9).

10.1 Mehrprozessorbetrieb

Begriffsdefinition

Sie befinden sich im Mehrprozessorbetrieb, sobald Sie eine Koordinatorbaugruppe gesteckt haben, unabhängig davon, wieviele CPUs gesteckt sind.

10.1.1 Wann verwenden Sie den Mehrprozessorbetrieb?

- Wenn Ihr Anwenderprogramm zu umfangreich für eine CPU ist und Speicherplatz knapp wird, so verteilen Sie Ihr Programm auf mehrere CPUs.
- Wenn ein bestimmter Teil Ihrer Anlage besonders schnell bearbeitet werden soll, so trennen Sie den betreffenden Programmteil aus dem Gesamtprogramm heraus und lassen diesen von einer eigenen "schnellen" CPU bearbeiten.
- Wenn Ihre Anlage aus mehreren Teilen besteht, die gut voneinander abzugrenzen und damit relativ eigenständig zu steuern bzw. zu regeln sind, so lassen Sie Anlagenteil 1 von CPU 1, Anlagenteil 2 von CPU 2 usw. bearbeiten.

Beachten Sie zum Mehrprozessorbetrieb unbedingt die entsprechenden Informationen in Ihrem Systemhandbuch. Sie können Ihnen u. a. bei der Entscheidung helfen, welche CPUs Sie für Ihr Problem am besten einsetzen.

10.1.2 Welche Kommunikationsmechanismen gibt es?

- Für den zyklischen Austausch weniger binärer Daten zwischen den CPUs (oder zwischen CPUs und einigen Kommunikationsprozessoren stehen Ihnen die **"Koppelmerker"** zur Verfügung,
- Beim Austausch größerer Datenmengen (z. B. ganzer Datenbausteine) zwischen CPU 948, CPU 946/947, CPU 928B, CPU 928 und CPU 922 werden Sie unterstützt durch die **"Sonderfunktionen für die Mehrprozessorkommunikation"** OB 200 bis OB 205 (mehr Information hierzu finden Sie ab Abschnitt 10.2).

10.1.3 Daten über Koppelmerker austauschen

Für den zyklischen Austausch binärer Daten stehen Ihnen die "Koppelmerker" zur Verfügung. Diese dienen in erster Linie zum **byteweisen** Übertragen von Informationen.

Dieser Datentransfer kann erfolgen zwischen

CPU(s)	und	CPU(s)
CPU(s)	und	Kommunikationsprozessor(en)

Das Systemprogramm überträgt die Koppelmerker einmal pro Zyklus. Bei einem Datentransfer zwischen CPUs werden die Koppelmerker physikalisch auf dem Koordinator zwischengespeichert.

Koppelmerker sind Merkerbytes, die transferiert werden. Sie werden für jede CPU im Datenbaustein DB 1 als Ein- oder Ausgangskoppelmerker definiert. Haben Sie z. B. das Merkerbyte 50 auf der CPU 1 als Koppelmerker**ausgang** definiert, so wird dessen Signalzustand zyklisch über den Koordinator zu der CPU übertragen, auf der das Merkerbyte 50 als Koppelmerker**eingang** definiert ist (siehe Abschnitt 10.1.5).

Hinweis
Es erfolgt **keine** Fehlermeldung, wenn das Koppelmerkerbyte zwar physikalisch vorhanden, aber nur einseitig beschrieben und nie ausgelesen wird oder umgekehrt.

Speicherbereich

Der Speicherbereich für die Koppelmerker auf dem Koordinator und den Kommunikationsprozessoren umfaßt die Adressen **F200H bis F2FFH**. Auf einer CPUbzw. einem Kommunikationsprozessor stehen Ihnen 256 Koppelmerkerbytes zur Verfügung.

Brückeneinstellung

Um Doppelbelegungen zu vermeiden, müssen Sie auf den Baugruppen KOR bzw. CP eine Rangierung vornehmen. Es können Blöcke zu je 32 Bytes ein- oder ausgeblendet werden. (In Ihrem Systemhandbuch finden Sie die Information, wie die Brücken einzustellen sind.)

Beispiel

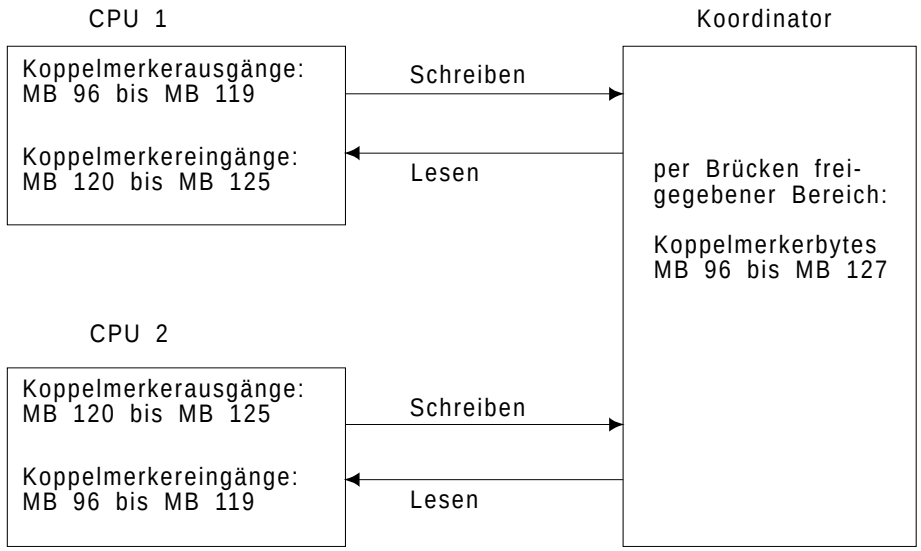


Bild 10-1 Übertragung der Koppelmerker im Mehrprozessorbetrieb

Hinweis

- Als Koppelmerker dürfen Sie nur diejenigen Merkerbytes angeben, die auf dem Koordinator oder dem (den) entsprechenden CP(s) freigegeben sind.

- Haben Sie auf einer oder mehreren CPUs ein bestimmtes Merkerbyte als Koppelmerker**eingang** definiert, so muß es auf einer anderen CPU oder einem anderen CP als Koppelmerker**ausgang** definiert sein. Und: Ein Merkerbyte darf nur auf **einer CPU als Koppelmerkerausgang** gekennzeichnet sein; Sie können es jedoch auf z. B. drei weiteren CPUs als Koppelmerkereingang definieren!

- Die auf einer CPU nicht als Koppelmerker definierten Merkerbytes können wie "normale" Merker verwendet werden!

S-Merker können Sie nicht als Koppelmerker verwenden!

Datenaustausch zwischen CPUs und Kommunikationsprozessoren

Sollen Daten zwischen einer CPU und einem Kommunikationsprozessor übertragen werden, müssen Sie die benötigte Anzahl der Koppelmerker auf dem Kommunikationsprozessor (CP) freigeben. Auch dort stehen Ihnen 256 byte zur Verfügung, die in Bereiche von 32 byte eingeteilt werden können.

Wenn von einer CPU Daten zu mehreren Kommunikationsprozessoren übertragen werden sollen, dürfen sich die auf den Kommunikationsprozessoren und dem Koordinator freigegebenen Bereiche **nicht überschneiden,** damit Adressen nicht mehrfach belegt werden.

Wenn Sie Koppelmerker **sowohl** auf dem Koordinator **als auch** auf einem oder mehreren Kommunikationsprozessoren benutzen wollen, so müssen Sie ebenfalls eine Doppeladressierung vermeiden:
Teilen Sie die Koppelmerker auf dem KOR und den CPs in Bereiche von je 32 Byte ein; diejenigen Koppelmerkerbytes, die Sie auf dem Kommunikationsprozessor verwenden, blenden Sie auf dem Koordinator durch Entfernen von Brücken aus (siehe Systemhandbuch).

Auch hier gilt, daß ein bestimmtes Merkerbyte nur auf **einer** CPU als Koppelmerkerausgang definiert werden darf. Hingegen kann ein bestimmtes Merkerbyte auf mehreren CPUs als Eingangskoppelmerker gekennzeichnet sein.

Beispiel

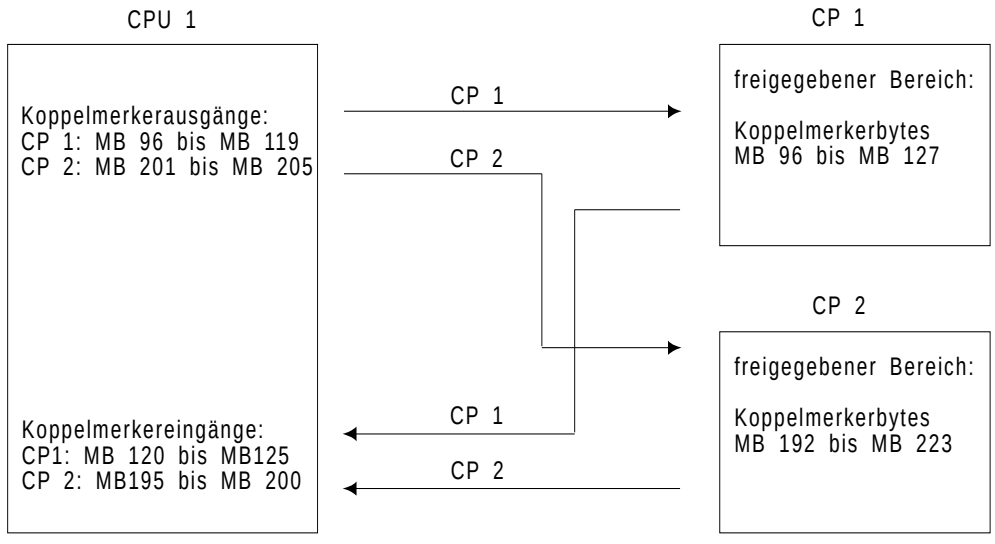


Bild 10-2 Beispiel für Kopelmerkerbereiche auf CPs

Übertragung der Koppelmerker im Mehrprozessorbetrieb

Die im DB 1 angegebenen Koppelmerkerbytes werden am Ende jedes Programmzyklus zusammen mit der Aktualisierung des Prozeßabbildes übertragen.

Der Koordinator erteilt reihum jeder CPU das Busfreigabesignal. Dabei darf eine CPU jeweils nur **ein** Byte übertragen. Dadurch kann es vorkommen, daß zusammengehörige Koppelmerkerinformationen von mehr als 1 Byte "auseinandergerissen" werden (verzahnte Übertragung).

Wenn Informationen übertragen werden sollen, die mehr als **ein** Byte umfassen, so können Sie durch eine entsprechende Parametrierung im Datenbaustein DX 0 erreichen, daß alle im DB 1 angegebenen Koppelmerker blockweise übertragen werden (siehe Kapitel 7). Solange eine CPU mit der Koppelmerkerübertragung beschäftigt ist, kann sie von einer anderen CPU nicht unterbrochen werden. Da die nächste CPU mit ihrer Übertragung warten muß, wird die Programmbearbeitung dieser CPU solange verzögert.

Mehrprozessorkommunikation

Für die Übertragung von Datenbausteinen, genauer: von Datenblöcken mit einer Größe von max. 64 byte (= 32 Datenwörtern), stehen Ihnen – in die CPU integriert – folgende Sonderfunktionen zur Verfügung:

- OB 200: INITIALISIEREN: Vorbesetzen
- OB 202: SENDEN: Senden eines Datenblocks
- OB 203: SENDE-TEST: Sendemöglichkeit testen
- OB 204: EMPFANGEN: Empfangen eines Datenblocks
- OB 205: EMPFANGS-TEST: Empfangsmöglichkeit testen

10.1.4 Peripherie- und Koppelmerkerzuteilung im Mehrprozessorbetrieb (DB 1)

Der Peripheriebereich des Automatisierungsgerätes ist auf dem S5-Bus nur **einmal** vorhanden. Der Peripheriebereich umfaßt die Adressen **F000H bis FFFFH**.

Im Mehrprozessorbetrieb greifen alle CPUs im Automatisierungsgerät "gleichzeitig" auf diesen Peripheriebereich zu. Damit dabei keine Daten überschrieben werden, muß der Peripheriebereich auf die einzelnen CPUs aufgeteilt werden.
Dazu müssen Sie **für jede CPU den DB 1** programmieren. Im DB 1 legen Sie fest, mit welchen **Ein- und Ausgängen** (Byteadressen 0-127) und mit welchen **Koppelmerkerein- und -ausgängen** die jeweilige CPU arbeiten soll.

Im Mehrprozessorbetrieb muß auch dann, wenn die CPU keine Peripherie oder Koppelmerker benutzt, ein (leerer) DB 1 vorhanden sein.

> **Hinweis**
> Nur die im DB 1 definierten Ein- und Ausgangsbytes werden bei der Aktualisierung des Prozeßabbilds durch die jeweilige CPU berücksichtigt!

10.1.5 Wie erstellen Sie den Datenbaustein DB 1?

Eingabe/Änderung des DB 1

- Erstellen/Ändern am PG über DB-1-Maske

  oder

- durch Editieren des DB 1 als Datenbaustein am PG und anschließendes Übertragen zur CPU.

> **Hinweis**
> Der eingegebene bzw. geänderte DB 1 wird nur über die Anlaufart NEUSTART der CPU ausgewertet!

Erstellen über DB-1-Maske

1. Wählen Sie an Ihrem PG den Editor für die DB-1-Maske (siehe Bild 10-3).
2. Tragen Sie die gewünschten Werte für "Digitale Eingänge" usw. als Dezimalzahlen ein.

3. Übernehmen Sie die eingetragenen Werte durch Betätigen der Übernahmetaste am PG.
   Das PG generiert danach den DB 1.

4. Übertragen Sie den DB 1 in die CPU oder laden Sie ihn in ein EPROM-Modul.

Hinweis
Die Zeitenblocklänge können Sie in der DX-0- und/oder DB-1-Maske angeben. Wir empfehlen Ihnen, diesen Parameter **nur in er DX-0-Maske anzugeben** (siehe Kapitel 7).

Beispiel für DB-1-Maske

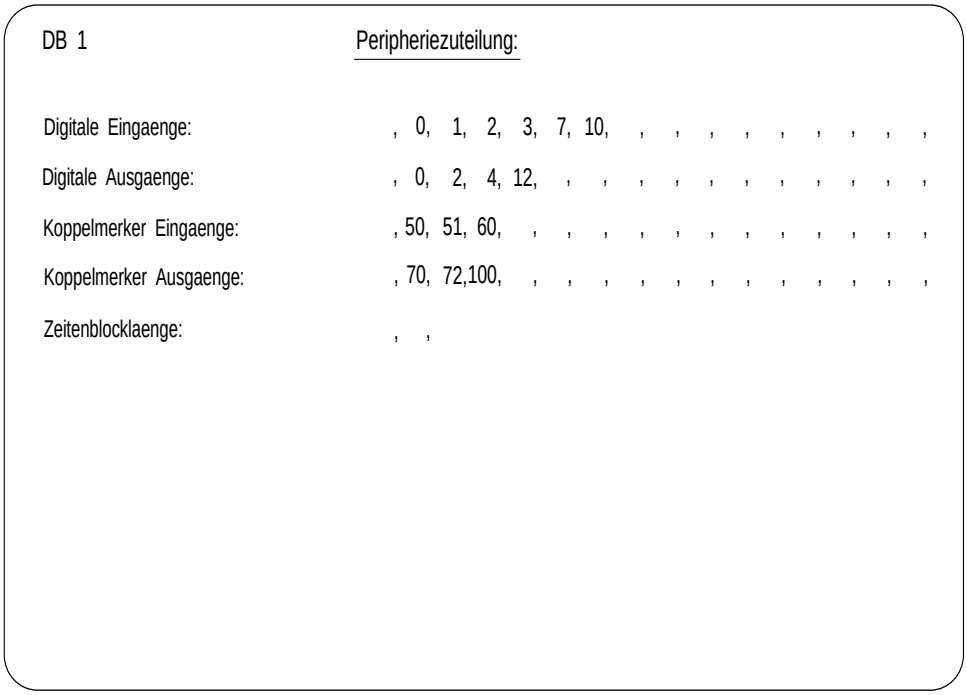


Bild 10-3 PG-Maske zum Generieren des DB 1

Editieren des DB 1 als Datenbaustein

1. Schreiben Sie in die Datenwörter 0, 1 und 2 die DB-1- Anfangskennung:

   DW 0: KH = 4D41 (’M’ ’A’)
   DW 1: KH = 534B (’S’ ’K’)
   DW 2: KH = 3031 (’0’ ’1’)

2. Geben Sie anschließend (ab Datenwort 3) die einzelnen Operandenbereiche ein. Vor jedem Operandenbereich müssen Sie eine bestimmte Kennung eingeben. Die möglichen Kennwörter sind:

Kennwort für digitale Eingänge	KH = DE00
Kennwort für digitale Ausgänge	KH = DA00
Kennwort für Eingangskoppelmerker	KH = CE00
Kennwort für Ausgangskoppelmerker	KH = CA00

   Anschließend an das Kennwort listen Sie die Nummern der verwendeten Ein- und Ausgänge im Festpunktformat auf.

3. Schließen Sie die Einträge mit der DB-1-Endekennung "KH = EEEE" ab und übertragen Sie den DB 1 in die CPU.

Hinweis
Die Reihenfolge der Einträge ist beliebig. Beachten Sie dabei, daß das Prozeßabbild der Ein- und Ausgänge in genau **der umgekehrten Reihenfolge** aktualisiert wird, in der Sie die **Adressen im DB 1** hinterlegt haben (d. h. der letzte Eintrag zuerst).
Mehrfacheinträge gleicher Bytes, z. B. für Testzwecke, sind möglich. Auch hier beachten Sie bitte, daß das Prozeßabbild mehrfach eingetragener Bytes mehrfach aktualisiert wird.

Beispiel für das Editieren des DB 1

```
DB1    FD: CPU928ST.S5D

  0:     KH = 4D41;           DW 0-2:
  1:     KH = 534B;                Anfangskennung
  2:     KH = 3031;                für DB 1
  3:     KH = DE00;           Kennwort für digitale Eingaenge
  4:     KF = +00000;         Eingangsbyte 0
  5:     KF = +00001;         Eingangsbyte 1
  6:     KF = +00002;         Eingangsbyte 2
  7:     KF = +00003;         Eingangsbyte 3
  8:     KF = +00007;              .
  9:     KF = +00010;         Eingangsbyte 10
 10:     KH = DA00;           Kennwort für digitale Ausgaenge
 11:     KF = +00000;         Ausgangsbyte 0
 12:     KF = +00002;         Ausgangsbyte 2
 13:     KF = +00004;              .
 14:     KF = +00012;         Ausgangsbyte 12
 15:     KH = CE00;           Kennwort für Koppelmerkereingaenge
 16:     KF = +00050;         Merkerbyte 50
 17:     KF = +00051;              .
 18:     KF = +00060;         Merkerbyte 60
 19:     KH = CA00;           Kennwort für Koppelmerkerausgaenge
 20:     KF = +00070;         Merkerbyte 70
 21:     KF = +00072;              .
 22:     KF = +00100;         Merkerbyte 100
 23:     KH = EEEE;           Endekennung
 24:
```

Übernahme des DB 1

Bei NEUSTART wird der DB 1 vom Systemprogramm übernommen. Es überprüft dabei, ob die im DB 1 angegebenen Ein- und Ausgänge bzw. Koppelmerker auf entsprechenden Baugruppen quittieren. Falls nicht, geht die CPU mit DB-1-Fehler in den Stoppzustand mit langsamem Blinken der STOP-LED. Ihr Anwenderprogramm wird dann nicht bearbeitet.

Sobald Sie einen DB 1 programmiert haben und dieser durch die Anlaufart NEUSTART von der CPU übernommen worden ist, gelten folgende Regeln:

- Zugriffe auf Peripheriebaugruppen über das Prozeßabbild sind nur für die im DB 1 angegebenen Ein- und Ausgänge möglich (Operationen L.../T... ...EB, ...EW, ...ED, ...AB, ...AW, ...AD und Verknüpfungsoperationen mit Ein- und Ausgängen). Zugriffe auf nicht im DB 1 eingetragene Prozeßabbild-Adressen führen zu einem Adressierfehler.

- Direktes **Laden** von Peripheriebytes unter Umgehung des Prozeßabbildes mit den Operatioenen L PY, L PW, L QB, L QW ist für alle quittierenden Eingänge – unabhängig von einem Eintrag im DB 1 – möglich.

- Direkter **Transfer** (T PY, T PW) auf Bytes von 0 bis 127 ist nur für die im DB 1 angegebenen Ausgänge möglich, da beim Direkttransfer zusätzlich das Prozeßabbild beschrieben wird. Schreibende Zugriffe auf nicht im DB 1 eingetragene Peripherie-Adressen führen zu einem Adressierfehler.

- **Transfer ohne Prozeßabbild** :
  Direkter Transfer auf Byteadressen **>127** ist **unabhängig von einem Eintrag im DB 1** möglich.
  Direkter Transfer auf Byteadressen des Erweiterten Peripheriebereichs (T QB, T QW) ist ebenfalls unabhängig von einem Eintrag im DB 1 möglich.

10.2 Mehrprozessorkommunikation

Begriffsbestimmung

Unter Mehrprozessor**kommunikation** versteht man den Austausch größerer Datenmengen (Datenbausteine) zwischen CPUs, die im Mehrprozessorbetrieb arbeiten. Für die Mehrprozessorkommunikation ist der Koordinator KOR 923C erforderlich.

10.2.1 Einführung

Für die Übertragung von Datenbausteinen, genauer: von Datenblöcken mit einer Größe von max. 64 byte (= 32 Datenwörtern), stehen Ihnen – in die CPU integriert – folgende Sonderfunktionen zur Verfügung:

- OB 200: INITIALISIEREN: Vorbesetzen
- OB 202: SENDEN: Senden eines Datenblocks
- OB 203: SENDE-TEST: Sendemöglichkeit testen
- OB 204: EMPFANGEN: Empfangen eines Datenblocks
- OB 205: EMPFANGS-TEST: Empfangsmöglichkeit testen

Die Sonderfunktions-OBs OB 200 und OB 202 bis OB 205 werden in den nachfolgenden Abschnitten vereinfacht "Kommunikations-OBs" genannt.

Vorausgesetzte Kenntnisse

Für die Anwendung dieser Funktionen genügen Grundkenntnisse über die Programmiersprache STEP 5 und über die Arbeitsweise von SIMATIC S5-Automatisierungsgeräten. Diese Grundkenntnisse werden in den im Literaturverzeichnis aufgeführten Schriften vermittelt.

Prinzipieller Ablauf

Um Daten zu übertragen, müssen Sie auf der Sende-CPU die Funktion SENDEN aktivieren und auf der Empfangs-CPU die Funktion EMPFANGEN.
Dabei werden aufeinanderfolgende Datenwörter eines DB- oder DX-Datenbausteins, die sich in der Sende-CPU befinden, über den Koordinator KOR 923C zur Empfangs-CPU transportiert und dort im DB- bzw. DX-Datenbaustein mit derselben Nummer und unter derselben Datenwort-Adresse abgelegt; d. h. es handelt sich um ein "1:1"-Kopieren.

Übertragungseinheit

Die Datenmenge, welche mittels der Funktionen SENDEN bzw. EMPFANGEN in einem Block übertragen werden kann, beträgt normalerweise 32 Wörter.
Falls die Baustein-Länge (ohne Kopf) kein Vielfaches von 32 Wörtern beträgt, so werden beim letzten Block ausnahmsweise weniger als 32 Wörter übertragen.

Der Datenbaustein in der Empfangs-CPU kann länger oder kürzer sein als der Sende-Datenbaustein. Entscheidend ist, daß die von der Funktion SENDEN übertragenen Datenwörter im Empfangs-Datenbaustein existieren; andernfalls erkennt die Funktion EMPFANGEN einen Fehler.

Beispiel:

	Sende-Daten in der Sende-CPU:	**Empfangs-Daten in der Empfangs-CPU:**
Daten-Baustein:	DB 17	DB 17
Datenwort-Adresse:	DW 32 bis DW 63	DW 32 bis DW 63

10.2.2 Wie Sender und Empfänger identifiziert werden

Jeder Datenblock, der unter den CPUs ausgetauscht wird, wird mit einer Nummer der Sende- und einer Nummer der Empfangs-CPU gekennzeichnet.
Die CPUs werden derart durchnumeriert, daß die am weitesten links gesteckte CPU die Nummer 1 erhält, die nach rechts folgenden CPUs erhalten eine jeweils um eins erhöhte Nummer.

Beispiel

AG 135U/155U:

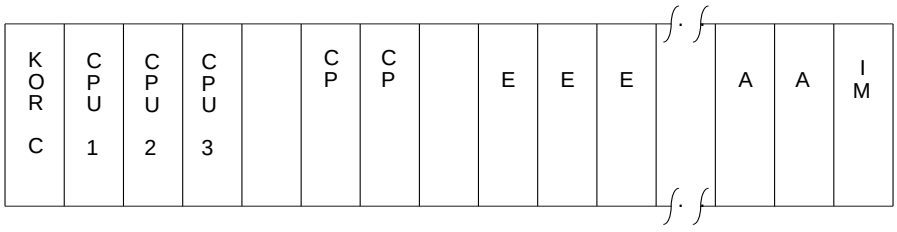


Bild 10-4 Sender-/Empfänger-Identifikation

10.2.3 Warum Daten zwischengespeichert werden

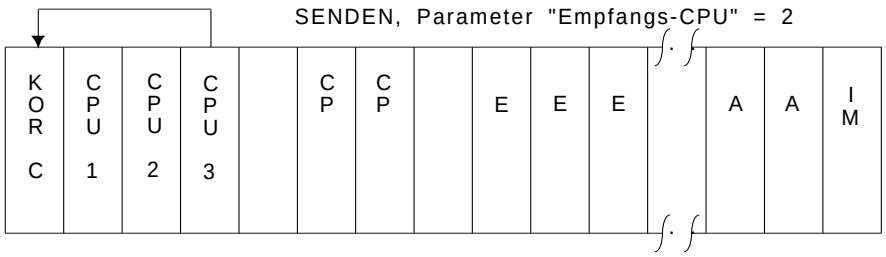
In der Regel wird der Mehrprozessorbetrieb dazu benutzt, zu bearbeitende Aufgaben auf mehrere CPUs zu verteilen. Da diese Aufgaben nicht identisch sind und zudem die Leistungsmerkmale der beteiligten CPUs unterschiedlich sein können, läuft im Mehrprozessorbetrieb die Programmbearbeitung auf den einzelnen CPUs immer **asynchron** ab. Dies bedeutet, daß die Daten einer sendenden CPU nicht sofort von der empfangenden CPU entgegengenommen werden können.

Aus diesem Grunde werden die zu übertragenden Daten im Koordinator KOR 923C zwischengespeichert. Die Nummer der "eigenen" CPU sowie die Nummer des Empfängers beim Senden bzw. die Nummer des Senders beim Empfangen legen dabei Quelle bzw. Ziel eines Datenblocks fest.

Beispiel

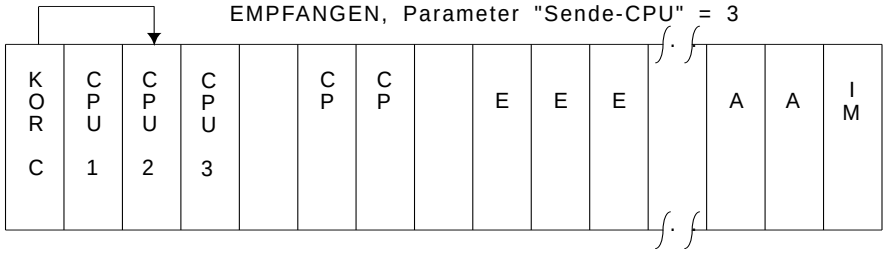
Datenübertragung von CPU 3 nach CPU 2:

1. Schritt:



Die CPU 3 legt ihre Daten im Zwischenspeicher des Koordinators ab.

2. Schritt:



Wenn CPU 2 empfangsbereit ist, kopiert sie die Daten aus dem Zwischenspeicher des Koordinators in den Ziel-DB.

10

10.2.4 Wie der Zwischenspeicher abgearbeitet und verwaltet wird

Abarbeitung

Dem Zwischenspeicher liegt das FIFO-Prinzip zugrunde (First in – first out, Warteschlangenprinzip). Somit ist die Empfangsreihenfolge gleich der Sendereihenfolge. Dies gilt für jede einzelne Verbindungsstrecke (gekennzeichnet durch Sende - und Empfangs-CPU) und ist von anderen Verbindungen unabhängig.

Datensicherung

Der Zwischenspeicher ist batteriegepuffert; damit ist der "Automatische Wiederanlauf nach Netzausfall" uneingeschränkt möglich. Durch einen Netzausfall während einer "laufenden" Datenübertragung gehen im Automatisierungsgerät keine Daten verloren.

Verwaltung

Die maximale Speicherkapazität des Koordinators KOR 923C beträgt 48 Speicherböcke mit einer festen Länge von je 32 Wörtern. Die Funktion INITIALISIEREN teilt diese Speicherblöcke den einzelnen Verbindungsstrecken zu.
Jeder **Speicherblock** nimmt genau einen **Datenblock** auf. Seine Länge kann 1 Datenwort bis 32 Datenwörtern betragen. Ein **Datenblock** wird von einer SENDEN-Funktion in einen **Speicherblock** eingetragen und von einer EMPFANGEN-Funktion wieder ausgetragen.
Die Anzahl der für eine Verbindungsstrecke vergebenen Speicherblöcke steht in direktem Zusammenhang mit den Parametern "Sende-Kapazität" (Funktion SENDEN, SENDE-TEST) und "Empfangs-Kapazität" (Funktion EMPFANGEN, EMPFANGS-TEST).

Die **Sende-Kapazität** gibt an, wieviele der für eine Verbindungsstrecke reservierten Speicherblöcke zu einem bestimmten Zeitpunkt frei sind.

Die **Empfangs-Kapazität** gibt an, wieviele der für eine Verbindungsstrecke reservierten Speicherblöcke zu einem bestimmten Zeitpunkt belegt sind.

Die Summe aus Sende- und Empfangs-Kapazität ist zu jedem Zeitpunkt gleich der Anzahl der für eine Verbindungsstrecke vergebenen Speicherblöcke.

Beispiel

Belegung des Zwischenspeichers durch eine Verbindungsstrecke:

Die Verbindungsstrecke "von CPU 3 nach CPU 2" wird initialisiert. Dabei werden ihr auf dem Zwischenspeicher des Koordinators sieben Speicherblöcke zugewiesen. Anschließend könnte z. B. folgende Datenübertragung ablaufen:

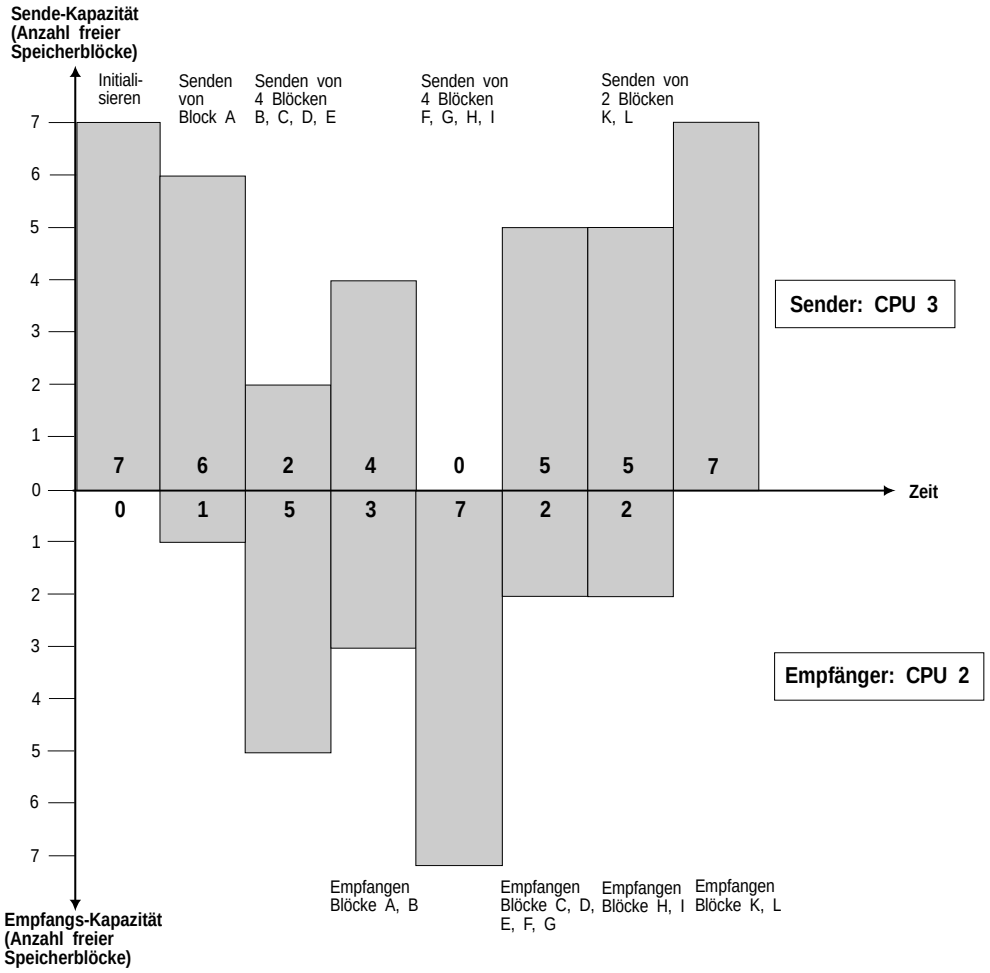


Bild 10-5 Beispiel für die Belegung des KOR-Zwischenspeichers

Senden und Empfangen von n Datenblöcken bedeutet, daß die entsprechenden Funktionen n-mal nacheinander aufgerufen werden.

Um eine einfachere Darstellung zu erreichen, wird in diesem Beispiel zunächst entweder gesendet oder empfangen.
Das gleichzeitige Senden (CPU 3) und Empfangen (CPU 2) ist jedoch auch möglich und sinnvoll ("Parallelverarbeitung im Mehrprozessor-Automatisierungsgerät"). Im Beispiel werden während des Sendens der Datenblöcke K und L die Datenblöcke H und I empfangen.

Das Beispiel verdeutlicht die Warteschlangen-Organisation des Zwischenspeichers: Die zuerst gesendeten Datenblöcke (A,B,C...) werden zuerst empfangen A,B,C...).

Zusammenfassung

Das Zwischenspeichern im Koordinator KOR 923C hat die Aufgabe, die asynchronen Abläufe von Sende- und Empfangs-CPU und deren unterschiedliche Arbeitsgeschwindigkeiten auszugleichen.

Da die Kapazität des Zwischenspeichers begrenzt ist, sollte der Empfänger "häufig" und "regelmäßig" überprüfen, ob Daten gespeichert sind (Funktion EMPFANGS-TEST, Empfangs-Kapazität > 0) bzw. versuchen, gespeicherte Daten abzuholen (Funktion EMPFANGEN). Zweckmäßigerweise ist die Funktion EMPFANGEN solange wiederholt aufzurufen, bis die Empfangs-Kapazität Null wird. Ein solcher Ablauf bewirkt, daß die gesendeten Daten nicht lange zwischengespeichert bleiben, sondern dem Empfänger aktuell zur Verfügung stehen. Zusätzlich werden damit Speicherblöcke frei (die Sende-Kapazität steigt); ein Blockieren des Senders (d. h. die Sende-Kapazität ist "erschöpft", ist Null) wird verhindert.

Hinweis
Während die Empfangs-Kapazität Null den Idealzustand repräsentiert (alle gesendeten Daten vom Empfänger abgeholt), deutet die Sende-Kapazität Null auf **Projektierungsfehler** hin:

- die Funktion SENDEN wird zu häufig aufgerufen,
- die Funktion EMPFANGEN wird zu selten aufgerufen

  oder

- der Verbindungsstrecke sind zu wenig Speicherblöcke zugewiesen. Die Kapazität des Zwischenspeichers reicht nicht aus, ein vorübergehendes Ungleichgewicht zwischen Sende- und Empfangs-Häufigkeit zu kompensieren.

10.2.5 Was Sie beim System-Anlauf beachten müssen

Die Mehrprozessor-Kommunikation erfordert, daß bei allen beteiligten CPUs der STOP-RUN-Übergang (= ANLAUF) **gleich** abläuft, d. h. entweder einheitlich NEUSTART oder einheitlich WIEDERANLAUF.

Durch entsprechende

- Bedienung (Frontschalter, Programmiergerät),
- Parametrierung (DX 0)

  und/oder

- Programmierung (mittels des Sonderfunktions-Organisationsbausteins OB 223
  "Stopp bei uneinheitlicher Anlaufart im Mehrprozessorbetrieb")

muß die **einheitliche** Anlaufart zumindest bei den an der Kommunikation beteiligten CPUs sichergestellt werden (vergl. Abschnitt 10.1.7).

NEUSTART

Im Organisationsbaustein OB 20 (NEUSTART) ist von **nur einer** CPU mittels der Funktion INITIALISIEREN der Zwischenspeicher (im KOR 923C) einzurichten. Hierbei werden eventuell noch vorhandene Daten zerstört.
Anschließend, also noch im ANLAUF, können Sie in den einzelnen CPUs die Funktionen SENDEN, SENDE-TEST, EMPFANGEN, EMPFANGS-TEST aufrufen. Durch geeignete Programmierung müssen Sie gewährleisten, daß dies erst geschieht, nachdem die Initialisierung des Zwischenspeichers im Koordinator korrekt durchgeführt wurde.
Nach Beendigung des ANLAUFs, also im RUN, wird das Anwenderprogramm **von Anfang an** bearbeitet, d. h. der erste Befehl des OB 1 bzw. des FB 0 ausgeführt.

WIEDERANLAUF

In den Organisationsbausteinen OB 21 (MANUELLER WIEDERANLAUF) und OB 22 (AUTOMATISCHER WIEDERANLAUF) dürfen Sie die Funktion INITIALISIEREN **nicht** benützen. Der Aufruf der Funktionen SENDEN, SENDE-TEST, EMPFANGEN, EMPFANGS-TEST kann zu Schwierigkeiten führen; hierzu sind die Hinweise im nachfolgenden Abschnitten zu beachten.

Nach Beendigung des WIEDERANLAUFs, also im RUN, wird das Anwenderprogramm nicht von Anfang an bearbeitet, sondern an der **unterbrochenen Stelle** fortgesetzt. Die Unterbrechungsstelle kann sich z. B. innerhalb der Funktion SENDEN befinden.

10.2.6 Was Sie beim Aufrufen der Kommunikations-OBs beachten müssen

Gehen Sie folgendermaßen vor:

1. Rufen Sie die Funktion INITIALISIEREN nur im Neustart-Organisationsbaustein OB 20 auf einer CPU auf.

2. Rufen Sie die Funktionen SENDEN, SENDE-TEST, EMPFANGEN, EMPFANGS-TEST entweder **nur** innerhalb der zyklischen Programmbearbeitung oder **nur** innerhalb der zeitgesteuerten Programmbearbeitung auf.

Doppelaufruf

Abhängig von der Parametrierung des DX 0 ("Unterbrechung an Befehlsgrenzen") und der Art der Programmbearbeitung (WIEDERANLAUF, Unterbrechungsbehandlung, z. B. OB 26 bei Zykluszeitfehler) ist es möglich, daß eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN und EMPFANGS-TEST unterbrochen wird.
Falls eine an der Unterbrechungsstelle eingeschachtelte Anwenderschnittstelle ebenfalls eine der Funktionen SENDEN, SENDE-TEST, EMPFANGEN und EMPFANGS-TEST enthält, **so erkennen diese einen unzulässigen Aufruf** (Doppelaufruf) und signalisieren Ihnen dies durch eine Fehler-Anzeige (Fehler-Nr. 67, Abschnitt 10.2.8).

Parallelverarbeitung

Nachdem Sie das Vorbesetzen des Zwischenspeichers abgeschlossen haben (Funktion INITIALISIEREN), können Sie die Funktionen SENDEN, SENDE-TEST, EMPFANGEN, EMPFANGS-TEST in beliebiger Kombination und Parametrierung in allen CPUs gleichzeitig und parallel bearbeiten lassen.

Betrachtet man eine einzelne Verbindungsstrecke (z. B. von CPU 2 nach CPU 3), so können die Funktion SENDEN (CPU 2) und die Funktion EMPFANGEN (CPU 3) gleichzeitig bearbeitet werden: Während die CPU 2 weitere Datenblöcke zum Koordinator sendet, kann die CPU 3 bereits Datenblöcke aus dem Zwischenspeicher des Koordinators abholen.

Belegte Bereiche

Die Kommunikations-OBs benötigen keinen Arbeitsbereich (z. B. für die Zwischenspeicherung von Variablen) und schlagen keine Datenbausteine auf. Sie greifen selbstverständlich auf Bereiche zu, die Parameter beinhalten, wobei nur die als Ausgangsparameter gekennzeichneten verändert werden.

Ergebnisanzeigen

Die Ergebnisanzeigen (ANZ 1/ANZ 0, VKE usw.) werden von den Kommunikations-OBs beeinflußt. Nähere Information dazu entnehmen Sie bitte dem Abschnitt 10.2.8.

Veränderung der Akkus

- CPU 922, CPU 928, CPU 928B: Die Inhalte AKKU 1 bis AKKU 4 sowie die Registerinhalte werden von den Kommunikations-OBs nicht verändert.
- CPU 946/947, CPU 948: Alle Registerinhalte sowie AKKU 1, 2 und 3 bleiben gleich, verändert wird lediglich der AKKU 4.

10.2.7 Wie parametrieren Sie die Kommunikations-OBs?

Es gibt bei den Kommunikations-OBs folgende Parameterarten:

- Eingangsparameter,
- Ausgangsparameter

  und

- Aufrufparameter.

Ein- und Ausgangsparameter befinden sich in einem max. 10 byte großen **Datenfeld im M-Merkerbereich**. Das Datenfeld untergliedert sich in einen Bereich für **Eingangsparameter**, einen Bereich für **Ausgangsparameter.**

Eingangsparameter

Die Eingangsparameter legen fest, wie eine Funktion abgewickelt werden soll. Sie werden vollständig oder teilweise von den Kommunikations-OBs gelesen und ausgewertet, schreibende Zugriffe finden nicht statt.

Ausgangsparameter

Die Ausgangsparameter enthalten alle Informationen, die das aufrufende Programm über das Ergebnis eines abgegebenen Auftrags wissen muß z. B. Fehleranzeigen.
Sie werden vollständig oder teilweise von den Kommunikations-OBs beschrieben, lesende Zugriffe finden nicht statt.

Hinweis
Sie können für **alle** Kommunikationsfunktionen **einen Merkerbereich mit 10 Merker-Bytes** vorsehen. Die einzelnen Funktionen benötigen jedoch eine unterschiedliche Anzahl von Bytes. Diese sind bei den Einzelfunktionen (Abschnitt 10.4ff) aufgeführt.

Aufrufparameter

Als Aufrufparameter wird bei allen Kommunikations-OBs die Nummer des ersten Merker-Bytes im Datenfeld (= Zeiger auf Datenfeld) im AKKU-1-L übergeben. Zulässige Werte dafür sind 0 bis 246.

Beispiel

```
Datenfeld mit Parametern der Funktion EMPFANGEN
(OB 204)

   MB x + 0:  Sende-CPU                    Eingangsparameter
   MB x + 1:                               nicht belegt

   MB x + 2:  Anzeigenbyte                 Ausgangsparameter
   MB x + 3:  Empfangs-Kapazität           Ausgangsparameter

   MB x + 4:  Baustein-Kennung             Ausgangsparameter
   MB x + 5:  Baustein-Nummer              Ausgangsparameter

   MB x + 6:  Adresse des ersten           Ausgangsparameter
   MB x + 7:  empfangenen Datenwortes      Ausgangsparameter

   MB x + 8:  Adresse des letzten          Ausgangsparameter
   MB x + 9:  empfangenen Datenwortes      Ausgangsparameter

Dieses Beispiel verdeutlicht, daß die Nummer des ersten M-Merkerbytes im
Datenfeld nicht größer als (MB) 246 sein kann und darf, da andernfalls das bis
zu 10 byte große Parameterfeld die Grenzen des Merkerbereichs überschreiten
würde (MB 255).
```

10.2.8 Wie können Sie die Ausgangsparameter auswerten?

Die Ausgangsparameter geben u. a. Hinweise, ob eine Funktion überhaupt bearbeitet werden konnte; falls nein, so zeigen sie den Grund für den Funktionsabbruch an.

Ergebnisanzeigen

Die Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN, EMPFANGS-TEST beeinflußen die Ergebnisanzeigen (siehe Programmieranleitungen der jeweiligen CPUs, Allgemeine Hinweise zu den STEP-5-Operationen):

- die OV- und OS-Bits (Wortanzeigen) werden immer gelöscht,
- die OR-, STA-, ERAB-Bits (Bitanzeigen) werden immer gelöscht,
- VKE, ANZ 1 und ANZ 0 geben Auskunft, ob eine Funktion korrekt und vollständig abgearbeitet worden ist.

Tabelle 10-1 Anzeigen der Kommunikations-OBs

Anzeigen			Auswertung	Bedeutung
VKE	ANZ 1	ANZ 0		
0	0	0	SPB=	Funktion vollständig und korrekt abgearbeitet
1	0	0	SPB=	Funktion abgebrochen, Zeiger auf Datenfeld unzulässig (>246) Funktion abgebrochen aufgrund eines Initialisierungskonfliktes
1	0	1	SPB= und SPM=	Funktion abgebrochen aufgrund eines Fehlers (Fehler-Nr. 1 bis 9)
1	1	0	SPB= und SPP=	Funktion abgebrochen aufgrund einer Warnung (Warnungs-Nr. 1 oder 2)

In **den nachfolgenden Ausführungen wird vorausgesetzt, daß der Zeiger auf das Datenfeld einen korrekten Wert enthält.** Dann ist im ersten Byte der Ausgangsparameter die Abbruchursache in detailierter Form hinterlegt.

Anzeigenbyte

Bit-Nr.	7	6	5	4	3	2	1	0
	W	F	I	0	Nummer			

W = 1: Warnung

F = 1: Fehler

I = 1: Initialisierungskonflikt

Nummer: - einer Warnung
- eines Fehlers
- eines Initialisierungskonflikts

Das erste Byte im Feld der Ausgangsparameter (Anzeigenbyte) zeigt ebenfalls an, ob eine Funktion korrekt und vollständig abgearbeitet worden ist. Der Grund eines Funktionsabbruchs wird detaillierter dargestellt als in den Ergebnisanzeigen.
Unter der oben getroffenen Vereinbarung, daß zumindest der Zeiger auf das Datenfeld einen korrekten Wert enthält, ist dieses Byte **immer** relevant.

Ist die Funktion korrekt und vollständig abgearbeitet worden, so sind alle Bits gelöscht (= 0), und alle weiteren Ausgangsparameter sind ebenfalls relevant.

Ist die Funktion mit einer Warnung abgebrochen worden (Bit-Nr. 7 = 1), so ist nur noch die Anzeige der Sende-/Empfangs- Kapazität relevant, weitere Ausgangsparameter (falls vorhanden) sind unverändert.

Ist die Funktion mit einem Fehler (Bit-Nr. 6 = 1) oder einem Initialisierungskonflikt (Bit-Nr. 5 = 1) abgebrochen worden, so sind alle weiteren Ausgangsparameter unverändert.

Auswertung des Anzeigenbytes

Die Kennungen ’W’, ’F’ und ’I’ zeigen u.a. an, welche Bedeutung die Information "Nummer" hat.
Neben dieser bitweisen Auswertung ist es auch möglich, das gesamte Anzeigenbyte als vorzeichenlose Festpunktzahl zu interpretieren. Bei einer **byteweisen Interpretation** des Anzeigenbytes ergeben sich Nummerngruppen mit folgender Bedeutung:

Tabelle 10-2 Anzeigenbyte der Kommunikations-OBs/Nummerngruppen

Wertebereich	Bedeutung
0	Funktion korrekt und vollständig abgearbeitet
33 bis 42	Funktion abgebrochen aufgrund eines Initialisierungskonfliktes
65 bis 73	Funktion abgebrochen aufgrund eines Fehlers
129 bis 130	Funktion abgebrochen aufgrund einer Warnung

Die Fehler werden entsprechend der aufsteigenden Reihenfolge der Fehlernummern erkannt und angezeigt. Dies bedeutet, daß durchaus mehrere Fehler vorliegen können, obwohl (momentan) nur **ein** Fehler angezeigt wird. Die weiteren Fehler werden dann bei Folgeaufrufen angezeigt.

Beispiel

```
Die Funktion SENDEN zeigt Fehler an und wird nicht
durchgeführt. Wenn Sie nun Programm- und/oder
Parameteränderungen durchführen und die Funktion
SENDEN erneut einen Fehler mit höherer Nummer als
vorher anzeigt, so können Sie daraus schließen, daß
Sie einen von mehreren Fehlern beseitigt haben.
```

Initialisierungskonflikt

Ein Initialisierungskonflikt kann nur bei der Funktion INITIALISIEREN auftreten. Er erfordert eine Änderung in der Programmierung/Parametrierung.

Initialisierungskonflikt-Nummern (byteweise Auswertung des Anzeigenbytes):

Tabelle 10-3 Anzeigenbyte: Initialisierungskonflikt-Nummern

Anz.-Byte	Bedeutung
33	Die für die Mehrprozessorkommunikation benötigten Kacheln (Nr. 252 bis Nr. 255) sind nicht bzw. nicht vollständig vorhanden.
34	Die für die Mehrprozessorkommunikation benötigten Kacheln (Nr. 252 bis 255) sind defekt.
35	Der Parameter "Automatisch/Manuell" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Kennung "Automatisch/Manuell" ist kleiner 1, - die Kennung "Automatisch/Manuell" ist größer 2.
36	Der Parameter "Anzahl CPUs" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Anzahl der CPUs ist kleiner 2, - die Anzahl der CPUs ist größer 4.
37	Der Parameter "Baustein-Kennung" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Baustein-Kennung ist kleiner 1, - die Baustein-Kennung ist größer 2.
38	Der Parameter "Baustein-Nummer" ist unzulässig, da es sich um einen Datenbaustein mit spezieller Bedeutung handelt.Unterscheiden Sie folgende Fälle: - falls Baustein-Kennung = 1 : DB 0, DB 1, DB 2 - falls Baustein-Kennung = 2 : DX 0, DX 1, DX 2
39	Der Parameter "Baustein-Nummer" ist fehlerhaft, da der parametrierte Datenbaustein nicht existiert.
40	Der Parameter "Anfangsadresse der Zuordnungsliste" ist zu groß bzw. der Datenbaustein zu kurz.

Anz.-Byte	Bedeutung
Fortsetzung der Tabelle 10-3:	
41	Die Zuordnungsliste im Datenbaustein ist nicht korrekt aufgebaut.
42	Die Summe der vergebenen Speicherblöcke ist größer als 48.

Fehler

Ein aufgetretener Fehler erfordert eine Änderung in der Programmierung/Parametrierung.

Fehler-Nummern (byteweise Auswertung des Anzeigenbytes):

Tabelle 10-4 Anzeigenbyte: Fehler-Nummern

Anz.-Byte	Bedeutung
65	Der Parameter "Empfangs-CPU" (SENDEN, SENDE-TEST) ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der "Empfangs-CPU" ist größer 4, - die Nummer der "Empfangs-CPU" ist kleiner 1, - die Nummer der "Empfangs-CPU" ist gleich der "eigenen" Nummer.
66	Der Parameter "Sende-CPU" (EMPFANGEN, EMPFANGS-TEST) ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der "Sende-CPU" ist größer 4, - die Nummer der "Sende-CPU" ist kleiner 1, - die Nummer der "Sende-CPU" ist gleich der "eigenen Nummer."
67	Der Aufruf des Sonderfunktions-Organisationsbausteins ist fehlerhaft (SENDEN, EMPFANGEN, SENDE-TEST, EMPFANGS-TEST). Unterscheiden Sie folgende Fälle: - Folgefehler, da die Funktion INITIALISIEREN nicht aufgerufen oder mit Initialisierungskonflikt beendet wurde, - Doppelaufruf: Der Aufruf dieser Funktion (SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST) ist unzulässig, da in dieser CPU bereits in einer untergeordneten Bearbeitungsebene (z. B. zyklische Programmbearbeitung) eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGSTEST aufgerufen wurde , - die "eigene" CPU-Nummer ist fehlerhaft (Systemdaten zerstört); nach NETZ EIN/NETZ AUS wird die CPU-Nr. vom Systemprogramm erneut erzeugt.

Anz.-Byte	Bedeutung
Fortsetzung der Tabelle 10-4:	
68	Die Verwaltungsdaten (Warteschlangenverwaltung) der angewählten Verbindungsstrecken sind fehlerhaft; den Zwischenspeicher im Koordinator KOR 923C richten Sie mit Hilfe der Funktion INITIALISIEREN neu ein (SENDEN, EMPFANGEN, SENDE-TEST, EMPFANGS-TEST).
69	Der Parameter "Baustein-Kennung" (SENDEN) bzw. die vom Sender überlieferte Baustein-Kennung (EMPFANGEN) ist unzulässig. Unterscheiden Sie folgende Fälle: - die Baustein-Kennung ist kleiner 1, - die Baustein-Kennung ist größer 2.
70	Der Parameter "Baustein-Nummer" (SENDEN) bzw. die vom Sender überlieferte Baustein-Nummer (EMPFANGEN) ist unzulässig, da es sich um einen Datenbaustein mit spezieller Bedeutung handelt. Unterscheiden Sie folgende Fälle: - falls Baustein-Kennung = 1 : DB 0, DB 1, DB 2 - falls Baustein-Kennung = 2 : DX 0, DX 1, DX 2
71	Der Parameter "Baustein-Nummer" (SENDEN) bzw. die vom Sender überlieferte Baustein-Nummer (EMPFANGEN) ist fehlerhaft. Der parametrierte Datenbaustein existiert nicht.
72	Der Parameter "Block-Nummer" (SENDEN) ist fehlerhaft. Der Datenbaustein ist zu kurz bzw. die Block-Nummer zu groß.
73	Der Datenbaustein ist zu klein, um den vom Sender gelieferten Datenblock aufzunehmen (EMPFANGEN).

Warnung

Die Funktion konnte nicht durchgeführt werden; der Funktionsaufruf ist zu wiederholen, z. B. im nächsten Zyklus.

Warnungs-Nummern (byteweise Auswertung des Anzeigenbytes):

Tabelle 10-5 Anzeigenbyte: Warnungs-Nummern

Anz-Byte	Bedeutung
129	Die Funktion SENDEN kann keine Daten übergeben, da die Sende-Kapazität bereits beim Funktionsaufruf gleich Null war.
130	Die Funktion EMPFANGEN kann keine Daten übernehmen, da die Empfangs-Kapazität bereits beim Funktionsaufruf gleich Null war.

10.3 Laufzeiten der Kommunikations-OBs

"Laufzeit" ist die Bearbeitungszeit der Sonderfunktions-Organisationsbausteine.

Tabelle 10-6 Laufzeiten der Kommunikations-OBs

Sonderfunktions-OB					
Baustein-Name	**CPU 922**	**CPU 928**	**CPU 928B**	**CPU 946/ 947**	**CPU 948**
OB 200/ Initialisieren	230 ms	130 ms	130 ms	128 ms	90 ms
OB 202/ Senden	806 µs (294 µs Grundlast + 16 µs/Wort); 118 µs bei Warnung	666 µs (250 µs Grundlast + 13 µs/Wort); 115 µs bei Warnung	696 µs (280 µs Grundlast + 13 µs/Wort); 145 µs bei Warnung	762 µs (426 µs Grundlast + 21 µs/Doppelwort); 243 µs bei Warnung	542 µs (220 µs Grundlast + 19 µs/Doppelwort); 110 µs bei Warnung
OB 203/ Sendetest	72 µs	50 µs	80 µs	207 µs	115 µs
OB 204/ Empfangen	825 µs (281 µs Grundlast + 17 µs/Wort); 115 µs bei Warnung	660 µs (244 µs Grundlast + 13 µs/Wort); 98 µs bei Warnung	690 µs (274 µs Grundlast + 13 µs/Wort); 128 µs bei Warnung	772 µs (421 µs Grundlast + 22 µs/Doppelwort); 243 µs bei Warnung	506 µs (218 µs Grundlast + 18 µs/Doppelwort); 132 µs bei Warnung
OB 205/ Empfangs-Test	70 µs	48 µs	78 µs	223 µs	120 µs

Die Zeit(-dauer) zwischen Aufruf eines Bausteins und seinem Abschluß kann erheblich größer werden, falls er von höherprioren Tätigkeiten unterbrochen wird (z. B. Zeitzellenaktualisierung usw.).

Die in der Tabelle 10-6 genannten Laufzeiten ergeben sich unter der Voraussetzung, daß von vier gesteckten CPUs nur diejenige CPU auf den S5-Bus zugreift, deren Laufzeiten gemessen werden. Falls weitere CPUs den Bus intensiv nutzen, steigt insbesondere beim Senden/Empfangen die Laufzeit an.

Übertragungszeit

Ein wichtiges Leistungsmerkmal einer Verbindungsstrecke (z. B. von CPU 1 nach CPU 2) ist die gesamte Datenübertragungszeit. Sie setzt sich aus den folgenden drei Komponenten zusammen:

- Sende-Dauer (siehe Laufzeit),
- Dauer der Zwischenspeicherung (im Koordinator KOR 923C)

  und

- Empfangs-Dauer (siehe Laufzeit)

Wie lange die zu übertragenden Daten "unterwegs" sind, wird also wesentlich von der Dauer der Zwischenspeicherung und damit von der Struktur des Anwenderprogramms bestimmt (vergl. "Zwischenspeicherung der Daten").

10.4 Funktion INITIALISIEREN (OB 200)

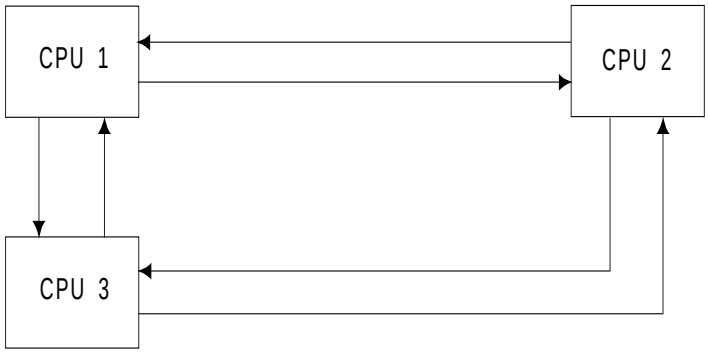
10.4.1 Funktion

Um von einer CPU an eine andere CPU Daten übertragen zu können, müssen diese vorübergehend zwischengespeichert werden. Die Funktion INITIALISIEREN richtet zu diesem Zweck einen Zwischenspeicher im Koordinator KOR 923C ein.
Die Speicherkapazität wird in Speicherblöcken mit einer festen Länge von 32 Wörtern eingerichtet (initialisiert).

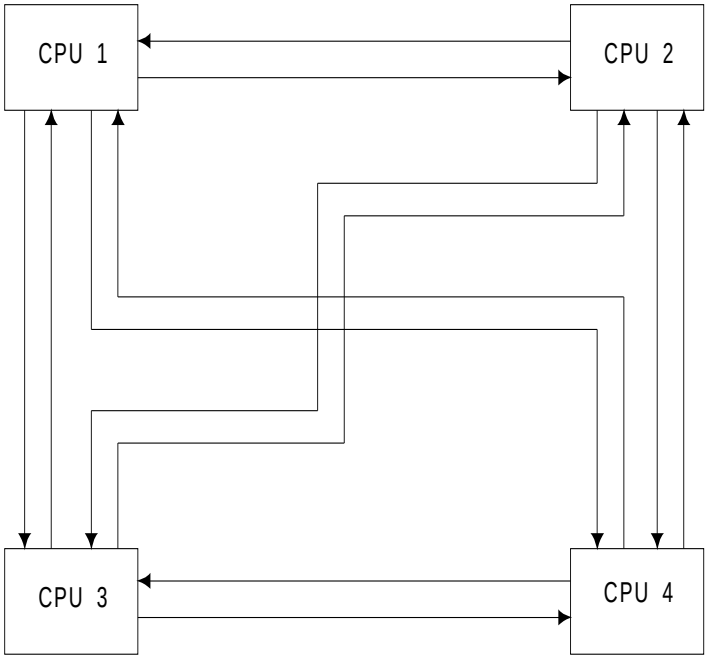
Jeder Speicherblock nimmt genau einen Datenblock auf, dessen Länge zwischen 1 Datenwort und 32 Datenwörtern liegen kann. Ein Datenblock wird von einer SENDEN-Funktion in einen Speicherblock eingetragen und von einer EMPFANGEN-Funktion wieder ausgetragen.

Bei zwei gesteckten CPUs ergeben sich zwei Verbindungsstrecken (Transferrichtungen, "Kanäle"):

Bei drei gesteckten CPUs ergeben sich sechs Verbindungsstrecken:



Bei vier gesteckten CPUs ergeben sich zwölf Verbindungsstrecken:



Mit der Funktion INITIALISIEREN wird festgelegt, wie die insgesamt **48** zur Verfügung stehenden Speicherblöcke den maximal zwölf Verbindungsstrecken zugeordnet werden.
D. h.: Jeder möglichen Verbindungsstrecke, gekennzeichnet durch die Parameter "Sende-CPU" und "Empfangs-CPU", steht eine bestimmte Speicherkapazität zur Verfügung.

Hinweis
Bevor auf den CPUs die Funktionen SENDEN / EMPFANGEN / SENDE-TEST / EMPFANGS-TEST aufgerufen werden dürfen, muß zuerst auf einer CPU die Funktion INITIALISIEREN aufgerufen und vollständig und fehlerfrei abgearbeitet worden sein.

Falls die Funktion INITIALISIEREN mehrfach nacheinander aufgerufen wird, so gilt die zuletzt parametrierte Zuordnung. Während die Funktion INITIALISIEREN von einer CPU bearbeitet wird, dürfen keine weiteren Funktionen der Mehrprozessorkommunikation, also auch nicht die Funktion INITIALISIEREN, auf anderen CPUs aufgerufen werden.

10.4.2 Aufrufparameter

Aufbau des (Paramerter-)Datenfeldes

Vor Aufruf des OB 200 müssen Sie im Datenfeld die Eingangsparameter bereitstellen. Der OB 200 benötigt im Datenfeld 8 M-Merkerbytes für Ein- und Ausgangsparameter:

MB x + 0:	Betriebsart (Automatisch/ Manuell)	Eingangsparameter
MB x + 1:	Anzahl CPUs	Eingangsparameter
MB x + 2:	Baustein-Kennung	Eingangsparameter
MB x + 3:	Baustein-Nummer	Eingangsparameter
MB x + 4:	Anfangsadresse der	Eingangsparameter
MB x + 5:	Zuordnungsliste	
MB x + 6:	Anzeigenbyte	Ausgangsparameter
MB x + 7:	Gesamt-Kapazität	Ausgangsparameter

AKKU-1-L

Bei Aufruf des OB 200 übergeben Sie in AKKU-1-L die Merkerbyte-Nr., an der das Parameterdatenfeld beginnt:

AKKU-1-LH:	0
AKKU-1-LL:	0 bis 246

10.4.3 Eingangsparameter

Betriebsart (Automatisch/Manuell)

Betriebsart = 1:	Automatisch
Betriebsart = 2:	Manuell
Betriebsart = 0 oder 3 bis 255:	Unzulässig, führt zu einem Initialisierungskonflikt

Anzahl der CPUs

Dieser Parameter ist nur relevant, wenn Sie die Betriebsart "Automatisch" gewählt haben. Bei Einstellung "Automatisch" werden die zur Verfügung stehenden Speicherblöcke **gleichmäßig** entsprechend der Anzahl der CPUs wie folgt aufgeteilt:

Anzahl der CPUs	Anzahl der Verbindungsstrecken	Speicherblöcke pro Verbindungsstrecke
2	2	24
3	6	8
4	12	4
0; 1; 5 bis 255	Unzulässig, führt zu einem Initialisierungs-Konflikt	

Baustein-Kennung, Baustein-Nummer. Adresse Zuordnungsliste

Die Parameter sind nur relevant, wenn Sie die Betriebsart "Manuell" gewählt haben. Sie müssen dann in einem Datenbaustein eine Zuordnungsliste bereitstellen, in der nach einem festgelegten Schema die 48 zur Verfügung stehenden Speicherblöcke (oder weniger) den maximal 12 Verbindungsstrecken zugeordnet werden. Diese Funktion ist insbesondere sinnvoll, falls nicht alle CPUs in gleichem Umfang Daten miteinander austauschen.
Den CPUs, die an der Mehrprozessorkommunikation nicht teilnehmen, brauchen und sollten Sie keine Speicherblöcke zuweisen.
Mit den Parametern

- Baustein-Kennung,
- Baustein-Nummer

  und

- Anfangsadresse der Zuordnungsliste

legen Sie fest, wo die Zuordnungsliste hinterlegt ist.

Baustein-Kennung

Kennung = 1:	DB-Datenbaustein
Kennung = 2:	DX-Datenbaustein
Kennung = 0 oder 3 bis 255 :	Unzulässig, führt zu einem Initialisierungskonflikt

Baustein-Nummer

Als Baustein-Nummer geben Sie die Nummer des Datenbausteins DB oder DX an, in dem die Zuordnungsliste liegt.

Anfangsadresse der Zuordnungsliste:

Sie ergibt zusammen mit der Baustein-Kennung und -Nummer den Bereich (genauer: die Anfangsadresse des Bereichs) im Datenbaustein, in dem die Zuordnungsliste hinterlegt ist.
Geben Sie als Adresse der Zuordnungsliste in den Merkerbytes MB x+4 (high Byte) und MB x+5 (low Byte) die Datenwortnummer an, bei der die Zuordnungsliste beginnt.

Zuordnungsliste

Mit der Zuordnungsliste legen Sie fest, wieviele der vorhandenen 48 Speicherblöcke den einzelnen Verbindungswegen zugeordnet werden sollen.

Die Liste wird vom Systemprogramm **nicht verändert.** Sie hat folgenden Aufbau:

Tabelle 10-7 Zuordnungsliste für OB 200 (Initialisieren)

Datenwort	Format	Wert	Bedeutung
DW n + 0	KC	S1	Sender = CPU 1
DW n + 1	KY	2 , **a**	Empfänger = CPU 2
DW n + 2	KY	3 , **b**	Empfänger = CPU 3
DW n + 3	KY	4 , **c**	Empfänger = CPU 4
DW n + 4	KC	S2	Sender = CPU 2
DW n + 5	KY	1 , **d**	Empfänger = CPU 1
DW n + 6	KY	3 , **e**	Empfänger = CPU 3
DW n + 7	KY	4 , **f**	Empfänger = CPU 4
DW n + 8	KC	S3	Sender = CPU 3
DW n + 9	KY	1 , **g**	Empfänger = CPU 1
DW n + 10	KY	2 , **h**	Empfänger = CPU 2
DW n + 11	KY	**4 , i**	Empfänger = CPU 4
DW n + 12	KC	S4	Sender = CPU 4
DW n + 13	KY	1 , **k**	Empfänger = CPU 1
DW n + 14	KY	2 , **l**	Empfänger = CPU 2
DW n + 15	KY	3 , **m**	Empfänger = CPU 3

Anstelle der Kleinbuchstaben a bis m (hier fettgedruckt) sind Zahlen zwischen 0 und 48 entsprechend den zugewiesenen Speicherblöcken einzusetzen; **ihre Summe darf 48 nicht überschreiten**.

Hinweis
Der in Tabelle 10-7 angegebene Aufbau muß eingehalten werden, auch wenn weniger als vier CPUs gesteckt sind.

Beispiel

Sie haben drei CPUs gesteckt. Von CPU 2 sind sehr viele Daten an die beiden anderen zu übertragen. Diese wiederum senden nur wenige Daten zurück an CPU 2 als Rückmeldung in einem logischen Quittungsverkehr. Zwischen **CPU 1 und CPU 3** ist **kein Datenaustausch** nötig.

Die Zuordnungsliste wird im Datenbaustein DB 40 ab DW 0 abgelegt und hat folgende Parameter:

```
DB40    FD:  CPU928ST.S5D

   0:     KC = S1;        Sender          CPU 1
   1:     KY = 2,2;       Empfänger:      CPU 2/2 Blöcke
   2:     KY = 3,0;       Empfänger:      CPU 3/kein Block
   3:     KY = 4,0;       Empfänger:      CPU 4 (nicht vorhanden)/kein Block
   4:     KC = S2;        Sender:         CPU 2
   5:     KY = 1,22;      Empfänger:      CPU 1/22 Blöcke
   6:     KY = 3,22;      Empfänger:      CPU 3/22 Blöcke
   7:     KY = 4,0;       Empfänger:      CPU 4 (nicht vorhanden)/kein Block
   8:     KC = S3;        Sender:         CPU 3
   9:     KY = 1,0;       Empfänger:      CPU 1/kein Block
  10:     KY = 2,2;       Empfänger:      CPU 2/2 Blöcke
  11:     KY = 4,0;       Empfänger       CPU 4 (nicht vorhanden)/kein Block
  12:     KC = S4;        Sender:         CPU 4 (nicht vorhanden)
  13:     KY = 1,0;       Empfänger:      CPU 1/kein Block
  14:     KY = 2,0;       Empfänger:      CPU 2/kein Block
  15:     KY = 3,0;       Empfänger:      CPU 3/kein Block
  16:
```

10.4.4 Ausgangsparameter

Anzeigenbyte

Dieses Byte informiert Sie, ob die Funktion INITIALISIEREN korrekt und vollständig abgearbeitet worden ist.

Initialisierungskonflikt

Die aufgeführten Initialisierungskonflikte werden entsprechend der aufsteigenden Reihenfolge ihrer Nummern von der Funktion erkannt und angezeigt.

Ein aufgetretener Initialisierungskonflikt erfordert eine Änderung in der Programmierung / Parametrierung.

Es können alle in der nachfolgenden Tabelle aufgeführten Nummern im Anzeigenbyte auftreten.

Anz.-Byte	Bedeutung
33	Die für die Mehrprozessorkommunikation benötigten Kacheln (Nr. 252 bis Nr. 255) sind nicht bzw. nicht vollständig vorhanden.
34	Die für die Mehrprozessorkommunikation benötigten Kacheln (Nr. 252 bis 255) sind defekt.
35	Der Parameter "Automatisch/Manuell" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Kennung "Automatisch/Manuell" ist kleiner 1, - die Kennung "Automatisch/Manuell" ist größer 2.
36	Der Parameter "Anzahl CPUs" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Anzahl der CPUs ist kleiner 2, - die Anzahl der CPUs ist größer 4.
37	Der Parameter "Baustein-Kennung" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Baustein-Kennung ist kleiner 1, - die Baustein-Kennung ist größer 2.
38	Der Parameter "Baustein-Nummer" ist unzulässig, da es sich um einen Datenbaustein mit spezieller Bedeutung handelt.Unterscheiden Sie folgende Fälle: - falls Baustein-Kennung = 1 : DB 0, DB 1, DB 2 - falls Baustein-Kennung = 2 : DX 0, DX 1, DX 2
39	Der Parameter "Baustein-Nummer" ist fehlerhaft, da der parametrierte Datenbaustein nicht existiert.
40	Der Parameter "Anfangsadresse der Zuordnungsliste" ist zu groß bzw. der Datenbaustein zu kurz.
41	Die Zuordnungsliste im Datenbaustein ist nicht korrekt aufgebaut.
42	Die Summe der vergebenen Speicherblöcke ist größer als 48.

10

Fehler

Eine Anzeige der Nummerngruppe "Fehler" kann bei der Funktion INITIALISIEREN nicht auftreten.

Warnung

Eine Anzeige der Nummerngruppe "Warnung" kann bei der Funktion INITIALISIEREN nicht auftreten.

Gesamt-Kapazität

Dieser Parameter gibt an, wieviel der zur Verfügung stehenden 48 Speicherblöcke den Verbindungsstrecken zugeordnet sind.
In der Betriebsart "Automatisch" wird dieser Parameter auf jeden Fall den Wert 48 enthalten. Bei der Betriebsart "Manuell" kann der Wert kleiner sein als 48. Dies bedeutet, daß vorhandene Speicherkapazität nicht genutzt wird.

10.5 Funktion SENDEN (OB 202)

10.5.1 Funktion

Die Funktion SENDEN übergibt einen Datenblock in den Zwischenspeicher des Koordinators KOR 923C. Zusätzlich zeigt sie an, wieviele Datenblöcke noch gesendet und zwischengespeichert werden können.

10.5.2 Aufrufparameter

Aufbau des (Paramerter-)Datenfeldes

Vor Aufruf des OB 202 müssen Sie im Datenfeld die Eingangsparameter bereitstellen. Der OB 202 benötigt im Datenfeld 6 M-Merkerbytes für Ein- und Ausgangsparameter:

MB x + 0:	Empfangs-CPU	Eingangsparameter
MB x + 1:	Baustein-Kennung	Eingangsparameter
MB x + 2:	Baustein-Nummer	Eingangsparameter
MB x + 3:	Block-Nummer	Eingangsparameter
MB x + 4:	Anzeigenbyte	Ausgangsparameter
MB x + 5:	Sende-Kapazität	Ausgangsparameter

AKKU-1-L

Bei Aufruf des OB 202 übergeben Sie in AKKU-1-L die Merkerbyte-Nr., an der das Parameterdatenfeld beginnt:

AKKU-1-LH:	0
AKKU-1-LL:	0 bis 246

10.5.3 Eingangsparameter

Empfangs-CPU

CPU-Nummer des Empfängers (Ziel); der zulässige Wert liegt zwischen 1 und 4, muß jedoch ungleich der "eigenen" Nummer sein.

Baustein-Kennung

Kennung = 1:	DB-Datenbaustein
Kennung = 2:	DX-Datenbaustein
Kennung = 0 oder 3 bis 255:	Unzulässig, führt zu einer Fehleranzeige

Baustein-Nummer

Die Baustein-Nummer ergibt zusammen mit der Baustein-Kennung und der Block-Nummer den Bereich, dem die Sende-Daten entnommen werden (und in dem sie auf der Empfangs-CPU abgelegt werden).

Hierbei ist zu beachten, daß bestimmte Datenbausteine besondere Bedeutungen haben, beispielsweise DB 0, DB 1 oder DX 0 (siehe Programmieranleitungen der jeweiligen CPUs). Diese Datenbausteine dürfen deshalb für die hier beschriebene Datenübertragung nicht verwendet werden!
Die Verwendung dieser Bausteinnummer führt zum Abbruch der Funktion mit Fehleranzeige.

Block-Nummer

Die Block-Nummer kennzeichnet den Quell-Datenbereich.

Block-Nummer	**Datenbereich**	
	erstes Datenwort	**letztes Datenwort**
0	DW 0	DW 31
1	DW 32	DW 63
2	DW 64	DW 95
3	DW 96	DW 127
4	DW 128	DW 159
5	DW 160	DW 191
6	DW 192	DW 223
7	DW 224	DW 255
8	DW 256	DW 287
9	DW 288	DW 319
:	:	:
:	:	:

Folgende Fälle sind zu unterscheiden:

- **DB ist länger als Quellbereich**:
  Ist der Datenbaustein ausreichend lang, so ergibt sich pro Block ein 32 Wörter großer Bereich gemäß der angeführten Tabelle.

- **DB ist zu kurz**:
  Liegt das Datenbaustein-Ende innerhalb des parametrierten Blokkes, so wird im letzten Block ein Bereich mit einer Länge zwischen einem und 31 Wörtern übertragen.

- **Block liegt außerhalb des DBs**:
  Ist die ermittelte erste Datenwort-Adresse eines Blocks bereits außerhalb der Datenbaustein-Länge, so wird von der Funktion SENDEN ein Fehler erkannt und angezeigt.

Beispiel

```
Ein Datenbaustein mit der Länge 80 Wörter: DW 0 bis
DW 74, 5 Wörter sind Bausteinkopf.

Blocknr:   erstes Datenw.: letztes Datenw.: Länge:

  0          DW  0              DW 31      32 Wörter
  1          DW 32              DW 63      32 Wörter

  2          DW 64              DW 74      11 Wörter

  3 und
  größer     fehlerhafte Parametrierung
```

10.5.4 Ausgangsparameter

Anzeigenbyte

Dieses Byte informiert Sie, ob die Funktion SENDEN korrekt und vollständig abgearbeitet worden ist.

Initialisierungskonflikt

Hat bei der Funktion SENDEN keine Bedeutung.

Fehler

Beim Aufrufen der Funktion SENDEN können folgende Fehlernummern (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	Bedeutung
65	Der Parameter "Empfangs-CPU" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der Empfangs-CPU ist größer 4, - die Nummer der Empfangs-CPU ist kleiner 1, - die Nummer der Empfangs-CPU ist identisch mit der "eigenen" Nummer.
67	Der Aufruf des Sonderfunktions-Organisationsbausteins ist fehlerhaft. Unterscheiden Sie folgende Fälle: - Folgefehler, da die Funktion INITIALISIEREN nicht aufgerufen oder mit Initialisierungskonflikt beendet wurde, - Doppelaufruf: Der Aufruf dieser Funktion SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST ist unzulässig, da in dieser CPU bereits in einer untergeordneten Bearbeitungsebene (z .B. zyklische Programmbearbeitung) eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST aufgerufen wurde, - die "eigene" CPU-Nummer ist fehlerhaft (Systemdaten zerstört); nach NETZ EIN/ NETZ AUS wird die CPU-Nr. vom Systemprogramm erneut erzeugt.
68	Die Verwaltungsdaten (Warteschlangenverwaltung) der angewählten Verbindungsstrecken sind fehlerhaft; der Zwischenspeicher im Koordinator KOR 923C ist mit Hilfe der Funktion INITIALISIEREN neu einzurichten.
69	Der Parameter "Baustein-Kennung" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Baustein-Kennung ist kleiner 1, - die Baustein-Kennung ist größer 2.
70	Der Parameter "Baustein-Nummer" ist unzulässig, da es sich um einen Datenbaustein mit spezieller Bedeutung handelt. Unterscheiden Sie folgende Sonderfälle: - falls Baustein-Kennung = 1 : DB 0, DB 1 - falls Baustein-Kennung = 2 : DX 0
71	Der Parameter "Baustein-Nummer" ist fehlerhaft. Der parametrierte Datenbaustein existiert nicht.
72	Der Parameter "Block-Nummer" ist fehlerhaft. Der Datenbaustein ist zu kurz bzw. die Block-Nummer zu groß.

Warnung

Die Funktion konnte nicht durchgeführt werden; der Funktionsaufruf ist zu wiederholen, z. B. im nächsten Zyklus.

Es kann folgende Warnungsnummer (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	**deutung**
129	Die Funktion SENDEN kann keine Daten übergeben, da die Sende-Kapazität (s. u.) bereits beim Funktionsaufruf gleich null war.

Sende-Kapazität

Im Parameter "Sende-Kapazität" wird angezeigt an, wieviele Datenblöcke gesendet und zwischengespeichert werden können.

10.6 Funktion SENDE-TEST (OB 203)

10.6.1 Funktion

Die Funktion SENDE-TEST ermittelt die Anzahl der freien Speicherblöcke im Zwischenspeicher des Koordinators KOR 923C. Entsprechend dieser Anzahl m kann die Funktion SENDEN m-mal aufgerufen werden um m Datenblöcke zu übergeben.

10.6.2 Aufrufparameter

Aufbau des (Paramerter-)Datenfeldes

Vor Aufruf des OB 203 müssen Sie im Datenfeld die Eingangsparameter bereitstellen. Der OB 203 benötigt im Datenfeld 4 M-Merkerbytes für Ein- und Ausgangsparameter:

MB x + 0:	Empfangs-CPU	Eingangsparameter
MB x + 1:	—	nicht belegt
MB x + 2:	Anzeigenbyte	Ausgangsparameter
MB x + 3:	Sende-Kapazität	Ausgangsparameter

AKKU-1-L

Bei Aufruf des OB 203 übergeben Sie in AKKU-1-L die Merkerbyte-Nr., an der das Parameterdatenfeld beginnt:

AKKU-1-LH:	0
AKKU-1-LL:	0 bis 246

10.6.3 Eingangsparameter

Empfangs-CPU

Die Nummer der "eigenen" CPU sowie die Nummer der Empfangs-CPU kennzeichnen diejenige Verbindungsstrecke, für die die Sende-Kapazität ermittelt wird.

10.6.4 Ausgangsparameter

Anzeigenbyte

Dieses Byte informiert Sie, ob die Funktion SENDE-TEST korrekt und vollständig abgearbeitet worden ist.

Initialisierungskonflikt

Hat bei der Funktion SENDE-TEST keine Bedeutung.

Fehler

Beim Aufrufen der Funktion SENDE-TEST können folgende Fehlernummern (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	Bedeutung
65	Der Parameter "Empfangs-CPU" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der Empfangs-CPU ist größer 4, - die Nummer der Empfangs-CPU ist kleiner 1, - die Nummer der Empfangs-CPU ist identisch mit der "eigenen" Nummer.
67	Der Aufruf des Sonderfunktions-Organisationsbausteins ist fehlerhaft. Unterscheiden Sie folgende Fälle: - Folgefehler, da die Funktion INITIALISIEREN nicht aufgerufen oder mit Initialisierungskonflikt beendet wurde, - Doppelaufruf: Der Aufruf dieser Funktion SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST ist unzulässig, da in dieser CPU bereits in einer untergeordneten Bearbeitungsebene (z .B. zyklische Programmbearbeitung) eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST aufgerufen wurde, - die "eigene" CPU-Nummer ist fehlerhaft (Systemdaten zerstört); nach NETZ EIN/NETZ AUS wird die CPU-Nr. vom Systemprogramm erneut erzeugt.
68	Die Verwaltungsdaten (Warteschlangenverwaltung) der angewählten Verbindungsstrecken sind fehlerhaft; der Zwischenspeicher im Koordinator KOR 923C ist mit Hilfe der Funktion INITIALISIEREN neu einzurichten.

Warnung

Eine Anzeige der Nummerngruppe "Warnung" kann bei der Funktion SENDE-TEST nicht auftreten.

Sende-Kapazität

Im Parameter "Sende-Kapazität" wird angezeigt an, wieviele Datenblöcke gesendet und zwischengespeichert werden können.

10.7 Funktion EMPFANGEN (OB 204)

10.7.1 Funktion

Die Funktion EMPFANGEN übernimmt einen Datenblock vom Zwischenspeicher des Koordinators KOR 923C. Zusätzlich zeigt sie an, wieviele Datenblöcke noch zwischengespeichert sind und empfangen werden können.
Die Funktion EMPFANGEN sollte in einer Schleife so oft aufgerufen werden, bis alle zwischengespeicherten Datenblöcke übernommen sind.

10.7.2 Aufrufparameter

Aufbau des (Paramerter-)Datenfeldes

Vor Aufruf des OB 204 müssen Sie im Datenfeld die Eingangsparameter bereitstellen. Der OB 204 benötigt im Datenfeld 10 M-Merkerbytes für Ein- und Ausgangsparameter:

MB x + 0:	Sende-CPU	Eingangsparameter
MB x + 1:	—	nicht belegt
MB x + 2:	Anzeigenbyte	Ausgangsparameter
MB x + 3:	Empfangs-Kapazität	Ausgangsparameter
MB x + 4:	Baustein-Kennung	Ausgangsparameter
MB x + 5:	Baustein-Nummer	Ausgangsparameter
MB x + 6:	Adresse des ersten	Ausgangsparameter
MB x + 7:	empfangenen Datenwortes	
MB x + 8:	Adresse des letzten	Ausgangsparameter
MB x + 9:	empfangenen Datenwortes	

AKKU-1-L

Bei Aufruf des OB 204 übergeben Sie in AKKU-1-L die Merkerbyte-Nr., an der das Parameterdatenfeld beginnt:

AKKU-1-LH:	0
AKKU-1-LL:	0 bis 246

10.7.3 Eingangsparameter

Sende-CPU

Der Empfangs-Baustein empfängt die Daten, die von der Sende-CPU geliefert wurden. Geben Sie die Nummer der Sende-CPU an. Der zulässige Wert liegt zwischen 1 und 4, muß jedoch ungleich der "eigenen" Nummer sein.

10.7.4 Ausgangsparameter

Anzeigenbyte

Dieses Byte informiert Sie, ob die Funktion EMPFANGEN korrekt und vollständig abgearbeitet worden ist.

Initialisierungskonflikt

Hat bei der Funktion EMPFANGEN keine Bedeutung.

Fehler

Beim Aufrufen der Funktion EMPFANGEN können folgende Fehlernummern (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	Bedeutung
66	Der Parameter "Sende-CPU" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der Empfangs-CPU ist größer 4, - die Nummer der Empfangs-CPU ist kleiner 1, - die Nummer der Empfangs-CPU ist identisch mit der "eigenen" Nummer.
67	Der Aufruf des Sonderfunktions-Organisationsbausteins ist fehlerhaft. Unterscheiden Sie folgende Fälle: - Folgefehler, da die Funktion INITIALISIEREN nicht aufgerufen oder mit Initialisierungskonflikt beendet wurde, - Doppelaufruf: Der Aufruf dieser Funktion SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST ist unzulässig, da in dieser CPU bereits in einer untergeordneten Bearbeitungsebene (z .B. zyklische Programmbearbeitung) eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST aufgerufen wurde, - die "eigene" CPU-Nummer ist fehlerhaft (Systemdaten zerstört); nach NETZ EIN/NETZ AUS wird die CPU-Nr. vom Systemprogramm erneut erzeugt.
68	Die Verwaltungsdaten (Warteschlangenverwaltung) der angewählten Verbindungsstrecken sind fehlerhaft; der Zwischenspeicher im Koordinator KOR 923C ist mit Hilfe der Funktion INITIALISIEREN neu einzurichten.
69	Die vom Sender übergebene Baustein-Kennung ist unzulässig. Unterscheiden Sie folgende Fälle: - die Baustein-Kennung ist kleiner 1, - die Baustein-Kennung ist größer 2.

Anz.-Byte	Bedeutung
Fortsetzung der Fehler-Nummern:	
70	Die vom Sender übergebene Baustein-Nummer ist unzulässig, da es sich um einen Datenbaustein mit spezieller Bedeutung handelt. Unterscheiden Sie folgende Fälle: - falls Baustein-Kennung = 1 : DB 0, DB 1 - falls Baustein-Kennung = 2 : DX 0
71	Die vom Sender übergebene Baustein-Nummer ist fehlerhaft. Der parametrierte Datenbaustein existiert nicht.
73	Der Datenbaustein ist zu klein, um den vom Sender gelieferten Datenblock aufzunehmen.

Warnung

Die Funktion konnte nicht durchgeführt werden; der Funktionsaufruf ist zu wiederholen, z. B. im nächsten Zyklus.

Es kann folgende Warnungsnummer (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	Bedeutung
130	Die Funktion EMPFANGEN kann keine Daten übernehmen, da die Empfangs-Kapazität bereits beim Funktionsaufruf gleich Null war.

Empfangs-Kapazität

Der Parameter "Empfangs-Kapazität" zeigt an, wieviel Datenblöcke noch zwischengespeichert sind und empfangen werden können.

Baustein-Kennung:

Kennung = 1:	DB-Datenbaustein
Kennung = 2:	DX-Datenbaustein
Kennung = 0 oder 3 bis 255:	Unzulässig, führt zu einer Fehleranzeige

Baustein-Nummer

Baustein-Nummer des DB/DX, in dem die Daten empfangen und abgelegt wordene sind (und aus dem sie in der Sende-CPU von der Funktion SENDEN entnommen worden sind).

Hierbei ist zu beachten, daß sich die Empfangs-Datenbausteine in einem Schreib-/Lese-Speicher (RAM) befinden müssen; die Verwendung von Nur-Lese-Speichern (EPROM) ist nur bei Sende-Datenbausteinen sinnvoll.

Adresse des ersten empfangenen Datenwortes

Datenwortnummer innerhalb des DB/DX, in der das erste übertragene/empfangene Datenwort abgelegt wurde.

Adresse des letzten empfangenen Datenwortes

Datenwortnummer innerhalb des DB/DX, in der das letzte übertragene/empfangene Datenwort abgelegt wurde.

Hinweis
Die Differenz zwischen den Adressen des ersten und des letzten übertragenen Datenwortes beträgt maximal 31, da pro Funktionsaufruf maximal 32 Wörter übertragen werden.

10.8 Funktion EMPFANGS-TEST (OB 205)

10.8.1 Funktion

Die Funktion EMPFANGS-TEST ermittelt die Anzahl belegter Speicherblöcke im Zwischenspeicher des Koordinators KOR 923C. Entsprechend dieser Anzahl m kann die Funktion EMPFANGEN m-mal aufgerufen werden um m Datenblöcke zu übernehmen.

10.8.2 Aufrufparameter

Aufbau des (Paramerter-)Datenfeldes

Vor Aufruf des OB 205 müssen Sie im Datenfeld die Eingangsparameter bereitstellen. Der OB 205 benötigt im Datenfeld 4 M-Merkerbytes für Ein- und Ausgangsparameter:

MB x + 0:	Sende-CPU	Eingangsparameter
MB x + 1:	—	nicht belegt
MB x + 2:	Anzeigenbyte	Ausgangsparameter
MB x + 3:	Empfangs-Kapazität	Ausgangsparameter

AKKU-1-L

Bei Aufruf des OB 204 übergeben Sie in AKKU-1-L die Merkerbyte-Nr., an der das Parameterdatenfeld beginnt:

AKKU-1-LH:	0
AKKU-1-LL:	0 bis 246

10.8.3 Eingangsparameter

Sende-CPU

Die Nummer der "eigenen" CPU sowie die Nummer der Sende-CPU kennzeichnen diejenige Verbindungsstrecke, für die die Empfangs-Kapazität ermittelt wird.

10.8.4 Ausgangsparameter

Anzeigenbyte

Dieses Byte informiert Sie, ob die Funktion EMPFANGS-TEST korrekt und vollständig abgearbeitet worden ist.

Initialisierungskonflikt

Hat bei der Funktion EMPFANGS-TEST keine Bedeutung.

Fehler

Beim Aufrufen der Funktion EMPFANGS-TEST können folgende Fehlernummern (byteweise Auswertung des Anzeigenbytes) auftreten:

Anz.-Byte	Bedeutung
66	Der Parameter "Sende-CPU" ist unzulässig. Unterscheiden Sie folgende Fälle: - die Nummer der Sende-CPU ist größer 4, - die Nummer der Sende-CPU ist kleiner 1, - die Nummer der Sende-CPU ist identisch mit der "eigenen" Nummer.
67	Der Aufruf des Sonderfunktions-Organisationsbausteins ist fehlerhaft. Unterscheiden Sie folgende Fälle: - Folgefehler, da die Funktion INITIALISIEREN nicht aufgerufen oder mit Initialisierungskonflikt beendet wurde, - Doppelaufruf: Der Aufruf dieser Funktion SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST ist unzulässig, da in dieser CPU bereits in einer untergeordneten Bearbeitungsebene (z .B. zyklische Programmbearbeitung) eine der Funktionen INITIALISIEREN, SENDEN, SENDE-TEST, EMPFANGEN oder EMPFANGS-TEST aufgerufen wurde, - die "eigene" CPU-Nummer ist fehlerhaft (Systemdaten zerstört); nach NETZ EIN/ NETZ AUS wird die CPU-Nr. vom Systemprogramm erneut erzeugt.
68	Die Verwaltungsdaten (Warteschlangenverwaltung) der angewählten Verbindungsstrecken sind fehlerhaft; der Zwischenspeicher im Koordinator KOR 923C ist mit Hilfe der Funktion INITIALISIEREN neu einzurichten.

Warnung

Eine Anzeige der Nummerngruppe "Warnung" kann bei der Funktion EMPFANGS-TEST nicht auftreten.

Empfangs-Kapazität

Der Parameter "Empfangs-Kapazität" zeigt an, wieviel Datenblöcke zwischengespeichert sind und empfangen werden können.

10.9 Anwendungen

Dieser Abschnitt erläutert Ihnen an Hand einiger Beispiele, wie Sie Ihre Mehrprozessor-Kommunikation programmieren können.

Hinweis
Wenn Sie die nachfolgend aufgeführten Funktionsbausteine verwenden und auf Ihrer CPU zusätzlich Alarme bearbeiten (z. B. durch OB 2), so achten Sie darauf, daß am Anfang einer Unterbrechungsbehandlung die "Schmiermerker" gerettet und am Ende wieder zurückgeschrieben werden.
Dies gilt auch bei der Einstellung "Unterbrechung an Bausteingrenzen", da der Aufruf der Sonderfunktions-Organisationsbausteine eine Bausteingrenze darstellt.

10.9.1 Aufruf der Sonderfunktions-OB über Funktionsbausteine

Die nachfolgend vorgestellten fünf Funktionsbausteine (FB 200 und FB 202 bis FB 205) enthalten den Aufruf des jeweiligen Sonderfunktions-Organisationsbausteins zur Mehrprozessorkommunikation (OB 200 und OB 202 bis OB 205).
Die Nummern der Funktionsbausteine sind frei gewählt und können geändert werden. Die Parameter der Sonderfunktions-OB werden bei Aufruf der Funktionsbausteine als Aktualparameter übergeben. Der direkte Aufruf der Sonderfunktions-Organisationsbausteine ist zwar laufzeitgünstiger, aber wegen der fehlenden Formalparameter schwieriger lesbar.

FB-Nummer	FB-Name	Funktion
FB 200	INITIAL	Vorbesetzen
FB 202	SENDEN	Senden eines Datenblockes
FB 203	SEND-TST	Sendemöglichkeit testen
FB 204	EMPFANG	Empfangen eines Datenblockes
FB 205	EMPF-TST	Empfangsmöglichkeit testen

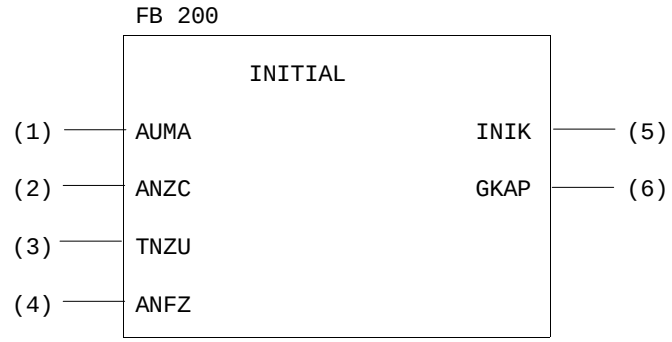
Der Merkerbereich von MB 246 bis maximal MB 255 wird von den Funktionsbausteinen als Parameterfeld für die Sonderfunktions-Organisationsbausteine benutzt.

Die genaue Bedeutung der Ein- und Ausgangsparameter können Sie der Beschreibung des verwendeten Sonderfunktions-Organisationsbausteins entnehmen.

Hinweis
Es handelt sich bei den folgenden Anwendungsbeispielen um fertige Applikationen, die Sie in Ihr Programm übernehmen können.

Programmieren der Funktionsbausteine

FB 200: Vorbesetzen der Verbindungen

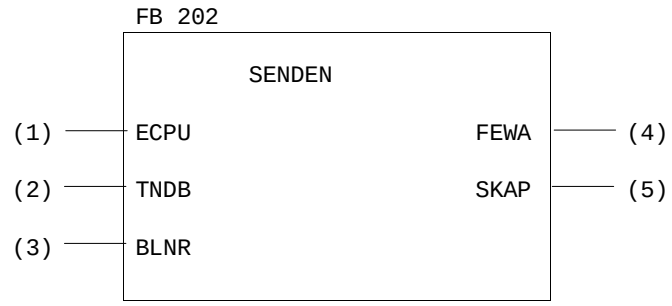


Parameter-Name	Bedeutung	Art	Typ	Parameterfeld
AUMA	**Au**tomatik/**Ma**nuell	E	BY	MB 246
ANZC	**Anz**ahl **C**PUs	E	BY	MB 247
TNZU	**T**yp (H-Byte) und **N**ummer (L-Byte) des Datenbausteins, der die **Zu**ordnungsliste enthält	E	W	MW 248
ANFZ	**Anf**angsadresse der **Z**uordnungsliste	E	W	MW 250
INIK	Initialisierungs-**K**onflikt	A	BY	MB 252
GKAP	**G**esamt-**Kap**azität	A	BY	MB 253

Fortsetzung auf der nächsten Seite

```
Fortsetzung von FB 200

FB 200                                                          LAE=45
NETZWERK 1          0000
NAME:INITIAL
BEZ    :AUMA     E/A/D/B/T/Z: E   BI/BY/W/D: BY
BEZ    :ANZC     E/A/D/B/T/Z: E   BI/BY/W/D: BY
BEZ    :TNZU     E/A/D/B/T/Z: E   BI/BY/W/D: W
BEZ    :ANFZ     E/A/D/B/T/Z: E   BI/BY/W/D: W
BEZ    :INIK     E/A/D/B/T/Z: A   BI/BY/W/D: BY
BEZ    :GKAP     E/A/D/B/T/Z: A   BI/BY/W/D: BY

0017      :L     =AUMA                      Automatisch/Manuell
0018      :T     MB 246
0019      :L     =ANZC                      Anzahl CPUs
001A      :T     MB 247
001B      :L     =TNZU                      DB-Typ, DB-Nr.
001C      :T     MW 248
001D      :L     =ANFZ                      Anfangsadresse der Zuordnungsliste
001E      :T     MW 250
001F      :
0020      :L     KB 246                     SF-OB:
0021      :SPA   OB 200                     "Initialisieren"
0022      :
0023      :L     MB 252                     Initialisierungs-Konflikt
0024      :T     =INIK
0025      :L     MB 253                     Gesamt-Kapazitaet
0026      :T     =GKAP
0027      :BE
```

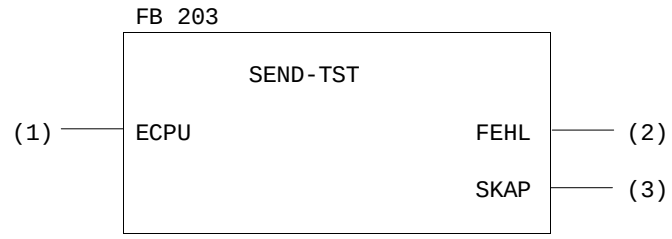
FB 202: Senden eines Datenblocks



Parameter-Name	Bedeutung	Art	Typ	Parameterfeld
ECPU	**E**mpfangs-**CPU**	E	BY	MB 246
TNDB	**T**yp (H-Byte) und **N**ummer (L-Byte) des Quell-**D**aten**b**austeins	E	W	MB 247
BLNR	**Bl**ock-**N**umme**r**	E	BY	MW 249
FEWA	**Fe**hler/**Wa**rnung	A	BY	MW 250
SKAP	**S**ende-**Kap**azität	A	BY	MB 251

```
FB 202                                                    LAE=40

NETZWERK 1       0000
NAME:SENDEN
BEZ   :ECPU   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :TNDB   E/A/D/B/T/Z: E     BI/BY/W/D: W
BEZ   :BLNR   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :FEWA   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :SKAP   E/A/D/B/T/Z: A     BI/BY/W/D: BY

0014      :L     =ECPU                    Empfangs-CPU
0015      :T     MB 246
0016      :L     =TNDB                    DB-Typ, DB-Nr.
0017      :T     MW 247
0018      :L     =BLNR                    Block-Nummer
0019      :T     MB 249
001A      :
001B      :L     KB 246                   SF-OB:
001C      :SPA   OB 202                   "Senden eines Datenblocks"
001D      :
001E      :L     MB 250                   Fehler/Warnung
001F      :T     =FEWA
0020      :L     MB 251                   Sende-Kapazitaet
0021      :T     =SKAP
0022      :BE
```

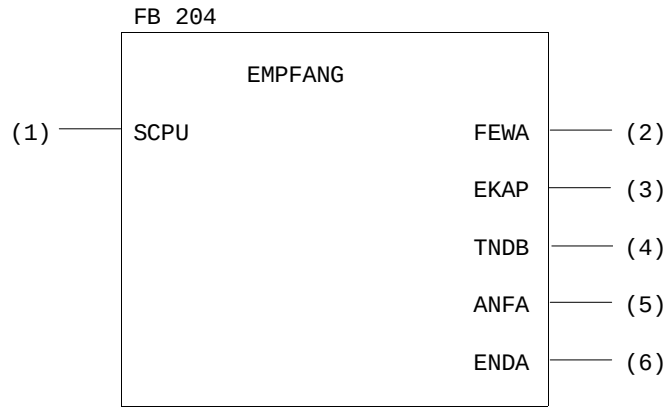
FB 203: Sendemöglichkeit testen



Parameter-Name	Bedeutung	Art	Typ	Parameter-Feld
ECPU	**E**mpfangs-**CPU**	E	BY	MB 246
FEHL	**Fehl**er	A	BY	MB 248
SKAP	**S**ende-**Kap**azität	A	BY	MB 249

```
FB 203                                                    LAE=30

NETZWERK 1       0000
NAME:SEND-TST
BEZ   :ECPU   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :FEHL   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :SKAP   E/A/D/B/T/Z: A     BI/BY/W/D: BY

000E     :L     =ECPU                      Empfangs-CPU
000F     :T     MB 246
0010     :
0011     :L     KB 246                     SF-OB:
0012     :SPA   OB 203                     "Sendemöglichkeit testen"
0013     :
0014     :L     MB 248                     Fehler
0015     :T     =FEHL
0016     :L     MB 249                     Sende-Kapazitaet
0017     :T     =SKAP
0018     :BE
```

FB 204: Empfangen eines Datenblocks



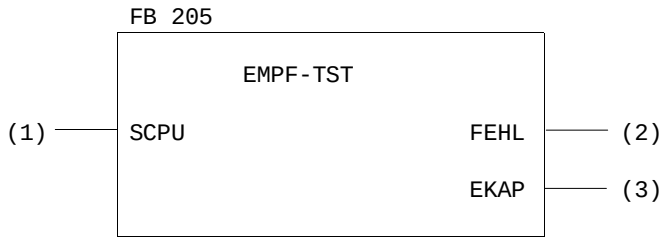
Parameter-Name	Bedeutung	Art	Typ	Parameter-Feld
SCPU	**S**ende-**CPU**	E	BY	MB 246
FEWA	**Fe**hler/**Wa**rnung	A	BY	MB 248
EKAP	**E**mpfangs-**Kap**azität	A	BY	MB 249
TNDB	**T**yp (H-Byte) und **N**ummer (L-Byte) des Ziel-**D**aten**b**austeins	A	W	MW 250
ANFA	Adresse des ersten empfangenen Datenwortes (**Anf**angs**a**dresse)	A	W	MW 252
ENDA	Adresse des letzten empfangenen Datenwortes (**Enda**dresse)	A	W	MW 254

Fortsetzung auf der nächsten Seite

Fortsetzung von FB 204:

```
FB 204                                                    LAE=45

NETZWERK 1       0000
NAME:EMPFANG
BEZ   :SCPU   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :FEWA   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :EKAP   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :TNDB   E/A/D/B/T/Z: A     BI/BY/W/D: W
BEZ   :ANFA   E/A/D/B/T/Z: A     BI/BY/W/D: W
BEZ   :ENDA   E/A/D/B/T/Z: A     BI/BY/W/D: W

0017      :L     =SCPU                      Sende-CPU
0018      :T     MB 246
0019      :
001A      :L     KB 246                     SF-OB:
001B      :SPA   OB 204                     "Empfangen eines Datenblocks"
001C      :
001D      :L     MB 248                     Fehler/Warnung
001E      :T     =FEWA
001F      :L     MB 249                     Empfangs-Kapazitaet
0020      :T     =EKAP
0021      :L     MW 250                     DB-Typ, DB-Nr.
0022      :T     =TNDB
0023      :L     MW 252                     Anfangsadresse
0024      :T     =ANFA
0025      :L     MW 254                     Endadresse
0026      :T     =ENDA
0027      :BE
```

FB 205: Empfangsmöglichkeit testen



Parameter-Name	Bedeutung	Art	Typ	Parameter-Feld
SCPU	**S**ende-**CPU**	E	BY	MB 246
FEHL	**Fehl**er	A	BY	MB 248
EKAP	**E**mpfangs-**Kap**azität	A	BY	MB 249

```
Fortsetzung von FB 205:

FB 205                                                    LAE=30

NETZWERK 1       0000
NAME:EMPF-TST
BEZ   :SCPU   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :FEHL   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :EKAP   E/A/D/B/T/Z: A     BI/BY/W/D: BY


000E     :L     =SCPU                      Sende-CPU
000F     :T     MB 246
0010     :
0011     :L     KB 246                     SF-OB:
0012     :SPA   OB 205                     "Empfangsmoeglichkeit testen"
0013     :
0014     :L     MB 248                     Fehler
0015     :T     =FEHL
0016     :L     MB 249                     Empfangs-Kapazitaet
0017     :T     =EKAP
0018     :BE
```

10.9.2 Übertragen von Datenbausteinen

Mit dem Funktionsbaustein UEBT-DAT (FB 110) soll in unserem Beispiel eine parametrierbare Anzahl von Datenblöcken aus einem Datenbaustein einer CPU in den Datenbaustein gleichen Typs und gleicher Nummer einer anderen CPU übertragen werden.
Die FB-Nummer (FB 110) ist zufällig gewählt und kann geändert werden.

Es wird zunächst das Programmieren des FB 110 und anschließend die Anwendung des FB 110 beschrieben.

Programmieren des FB 110

```
FB 110: Übertragen eines Datenbausteins

Aufgabenstellung

Der zu übertragende Datenbereich wird über die Eingangsparameter ERSB
(= Nummer des ersten zu übertragenden Datenblocks) und ANZB (= Anzahl der zu
übertragenden Datenblöcke) festgelegt. Ein Datenblock besteht normalerweise
aus 32 Datenwörtern. Je nach Datenbaustein-Länge werden beim letzten Daten-
block gegebenenfalls weniger als 32 Datenwörter übertragen.

Die Übertragung wird mit einer positiven Flanke am Starteingang STAR angestoßen.
Ist anschließend der Ausgangsparameter REST gleich Null, so konnte der
Funktionsbaustein UEBT-DAT alle Datenblöcke (gemäß Parameter ANZB) senden.


                                        Fortsetzung auf der nächsten Seite
```

Fortsetzung 1 von FB 110:

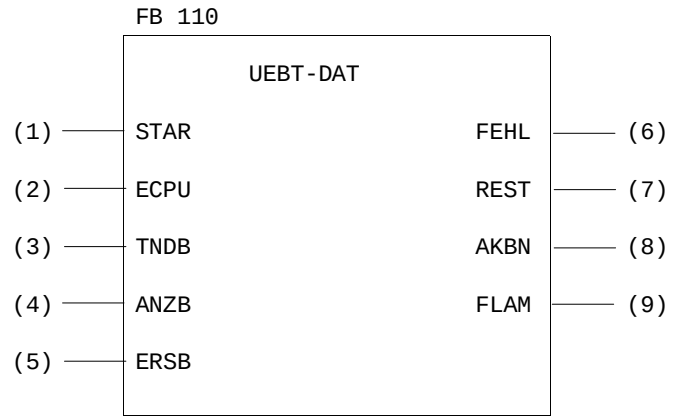
Enthält der Ausgangsparameter REST jedoch einen Wert größer als Null, sind – z. B. im nächsten Zyklus – Folgeaufrufe erforderlich. In diesem Fall darf der gesamte Parametersatz (d.h. die Werte aller Parameter) vom Anwender (-Programm) erst dann verändert werden, wenn der Ausgangsparameter REST den Wert Null hat, d. h. wenn die Datenübertragung abgeschlossen ist.

Der mehrfache Aufruf des Funktionsbausteins UEBT-DAT mit jeweils verschiedenen Parametersätzen ist möglich. Hierbei werden verschiedene Datenbereiche gleichzeitig ("ineinander verzahnt") übertragen. Zusätzlich können die Sonderfunktions-Organisationsbausteine zur Mehrprozessorkommunikation OB 202 bis OB 205 "direkt" eingesetzt werden. Von dieser Möglichkeit wird im Anwendungsbeispiel Gebrauch gemacht.

Falls innerhalb des Funktionsbausteins UEBT-DAT die Funktion SENDEN (OB 202) nicht korrekt abgearbeitet werden konnte, wird die jeweilige Fehlernummer im Ausgangsparameter FEHL übergeben, das VKE = ’1’ und der Ausgangsparameter REST = ’0’ gesetzt.

Der Funktionsbaustein UEBT-DAT verwendet die Merkerbytes MB 246 bis MB 251 als Schmiermerker. Alle anderen Variablen, deren Wert solange von Bedeutung ist, bis nach mehrfachem Aufruf des Funktionsbausteins UEBT-DAT der Ausgangsparameter REST = ’0’ ist, erhalten über den Mechanismus der Formal-/Aktual-Parameter Speicherplätze zugewiesen. Dieses Verfahren ist erforderlich, damit verschiedene Datenbausteine gleichzeitig übertragen werden können.

Realisierung



Fortsetzung auf der nächsten Seite

Fortsetzung 2 von FB 110:

Parameter-Name	Bedeutung	Art	Typ
STAR	Übertragung des Datenbausteins nach positiver Flanke **star**ten.	E	BI
ECPU	**E**mpfangs-**CPU**	E	BY
TNDB	**T**yp (H-Byte) und **N**ummer (L-Byte) des zu übertragenden **D**aten**b**austeins.	E	W
ANZB	**Anz**ahl der zu übertragenden Daten**b**löcke.	E	BY
ERSB	Nummer des **ers**ten zu übertragenden Daten**b**locks.	E	BY
FEHL	**Fehl**er	A	BY
REST	Anzahl der noch zu übertragenden Datenblöcke.	A	BY
AKBN [1)]	**Ak**tuelle **B**lock-**N**ummer	A	BY
FLAM [1)]	**Fla**nken**m**erker	A	BI

[1)] Interne Zwischenmerker, nicht zur Auswertung vorgesehen.

```
FB 110                                                    LAE=89

NETZWERK 1       0000
NAME:UEBT-DAT
BEZ   :STAR   E/A/D/B/T/Z: E     BI/BY/W/D: BI
BEZ   :ECPU   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :TNDB   E/A/D/B/T/Z: E     BI/BY/W/D: W
BEZ   :ANZB   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :ERSB   E/A/D/B/T/Z: E     BI/BY/W/D: BY
BEZ   :FEHL   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :REST   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :AKBN   E/A/D/B/T/Z: A     BI/BY/W/D: BY
BEZ   :FLAM   E/A/D/B/T/Z: A     BI/BY/W/D: BI

0020     :L     =ECPU                          Parameterfeld fuer SF-OB 202
0021     :T     MB 246                         vorbesetzen
0022     :L     =TNDB
0023     :T     MW 247
0024     :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 3 von FB 110:

```
0025      :L     =REST                    zuerst eventuell noch vorhandene
0026      :L     KB 0                     Bloecke aussenden
0027      :><F
0028      :SPB   =UEBT
0029      :
002A      :UN    =STAR                    positive Flanke
002B      :RB    =FLAM                    am START-Eingang ?
002C      :ON    =STAR
002D      :O     =FLAM
002E      :SPB   =GUT
002F      :S     =FLAM
0030      :
0031      :L     =ANZB                    die globalen Merker nach
0032      :T     =REST                    einer postiven Flanke am
0033      :L     =ERSB                    START-Eingang initialisieren
0034      :T     =AKBN
0035      :
0036      :L     =REST                    solange REST ><0 ist,
0038 SCHL:L      KF+0                     weiterhin versuchen,
0039      :!=F                            Datenbloecke auszusenden
003A      :SPB   =GUT
003B UEBT:L      =AKBN
003C      :T     MB 249
003D      :L     KB 246                   SF-OB:
003E      :SPA   OB 202                   "Senden eines Datenblocks"
003F      :L     MB 250
0040      :SPM   =FEHL                    Abbruch bei Fehler
0041      :SPP   =GUT                     Abbruch, falls Sende-Kap. = 0
0042      :L     =AKBN                    Block-Nummer
0043      :I     1                        inkrementieren
0044      :T     =AKBN
0045      :L     =REST                    Anzahl der verbleibenden
0046      :D     1                        Bloecke dekrementieren
0047      :T     =REST
0048      :SPA   =SCHL
0049      :
004A GUT  :U     M 0.0                    regulaeres Programmende:
004B      :UN    M 0.0
004C      :L     KB 0                     VKE = 0, FEHL = 0
004D      :T     =FEHL
004E      :BEA
004F      :
0050 FEHL:T      =FEHL                    Programmende bei Fehler:
0051      :L     KB 0
0052      :T     =REST                    VKE = 1, FEHL enthaelt Fehlernummer
0053      :BE
```

Anwendung des FB 110

Anwendung des FB 110

Aufgabenstellung

Die CPU 1 soll im zyklischen Anwenderprogramm die Datenbausteine DB 3 (Datenblöcke 2 bis 5) und DB 4 (Datenblöcke 1 bis 3) an die CPU 2 senden. In der CPU 2 soll ebenfalls im zyklischen Anwenderprogramm die Funktion EMPFANGEN (OB 04) aufgerufen werden.

Realisierung

Funktion	**CPU 1**	**CPU 2**
	aufgerufen in:	aufgerufen in:
Initialisieren (OB 200)	OB 20	–
Sende-Organisation (FB 1)	OB 1	–
Empfangs-Organisation (FB 2)	–	OB 1
	vorhanden:	vorhanden:
Sende-DB	DB 3; DB 4	–
Empfangs-DB	–	DB 3; DB 4

Das Anwenderprogramm im Funktionsbaustein FB 1 der CPU 1 enthält zweimal den Aufruf des Funktionsbausteins UEBT-DAT mit jeweils unterschiedlichen Parametersätzen.
Nach einer positiven Flanke am Eingang E 2.0 beginnt die Übertragung des ersten Datenbausteins DB 3. Eine positive Flanke am Eingang E 2.1 startet die Übertragung des zweiten Datenbausteins DB 4.

```
FB 1                                                        LAE=yy

NETZWERK 1        0000
NAME:S-ORG
0000     :L     KB 2                    an die CPU 2 ..
0001     :T     MB 0
0002     :L     KY 1,3                  .. aus dem Datenbaustein DB 3
0003     :T     MW 1
0004     :L     KB 4                    .. vier Datenbloecke
0005     :T     MB 3
0006     :L     KB 2                    .. ab Datenblock 2 senden
0007     :T     MB 4
0008     :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 1 des Anwendungsbeispiels:

```
0009      :SPA   FB 110
000A NAME :UEBT-DAT
000B STAR :      E 2.0
000C ECPU :      MB 0
000D TNDB :      MW 1
000E ANZB :      MB 3
000F ERSB :      MB 4
0010 FEHL :      MB 5
0011 REST :      MB 6
0012 AKBN :      MB 7
0013 FLAM :      M 8.0
0014      :
0015      :
0016      :SPB   =HALT                      Abbruch nach Fehler
0017      :
0018      :L     KB 2                       an die CPU 2 ..
0019      :T     MB 10
001A      :L     KY 1,4                     .. aus dem Datenbaustein DB 4
001B      :T     MW 11
001C      :L     KB 3                       .. drei Datenblöcke
001D      :T     MB 13
001E      :L     KB 1                       .. ab Datenblock 1 senden
001F      :T     MB 14
0020      :
0021      :SPA   FB 110
0023 NAME :UEBT-DAT
0024 STAR :      E 2.1
0025 ECPU :      MB 10
0026 TNDB :      MW 11
0027 ANZB :      MB 13
0028 ERSB :      MB 14
0029 FEHL :      MB 5
002A REST :      MB16
002B AKBN :      MB17
002C FLAM :      M 8.1
002D      :
002E      :
002F      :SPB   =HALT                      Abbruch nach Fehler
0030      :BEA
0031      :
0032 HALT :
0033      :                                 hier erfolgt die Fehlerbehandlung
0034      :                                 (z. B. Stopp, Meldungsausgabe
0035      :                                 auf Drucker, ...)
0036      :

00xx      :BE
```

Fortsetzung auf der nächsten Seite

Fortsetzung 2 des Anwendungsbeispiels:

In der CPU 2 überträgt die vom FB 2 aufgerufene Funktion EMPFANGEN (OB 204) jeden ausgesendeten Datenblock in den zugehörigen Datenbaustein. Der vollständige Empfang eines Datenbausteins kann sich über mehrere Zyklen verteilen.

```
FB 2                                                        LAE=yy

NETZWERK 1       0000
NAME:E-ORG
0000      :L     KB 1                      Daten von CPU 1 empfangen
0001      :T     MB 246
0002      :
0003 SCHL:L      KB 246                    SF-OB:
0004      :SPA   OB 204                    "Empfangen"
0005      :SPM   =FEHL                     Abbruch bei Fehler
0006      :L     MB 249                    Die Funktion "Empfangen"
0007      :L     KB 0                      wird solange aufgerufen,
0008      :><F                             bis der Zwischenspeicher
0009      :SPB   =SCHL                     keine weiteren Datenbloecke
000A      :                                mehr enthaelt, d. h. die
000B      :BEA                             Empfangskapazitaet = 0 ist.
000C FEHL:
000D      :                                hier erfolgt die Fehlerbehandlung
000E      :                                (z. B. Stopp, Meldungsausgabe
000F      :                                auf Drucker, ...)

00xx      :BE
```

10.9.3 Erweiterung des Koppelmerkerbereichs

Problemstellung

Im Automatisierungsgerät S5-135U/155U kann jedes der 256 Merkerbytes einer CPU durch Eintrag in den Datenbaustein DB 1 zum Eingangs- oder Ausgangs-Koppelmerker werden. Dadurch verringert sich jedoch die Anzahl der "normal" verwendbaren Merkerbytes. Weiterhin sind zur Übertragung eines Datensatzes (mehrere Bytes) zusätzliche Maßnahmen (Semaphor-Variable oder DX-0-Parametrierung "Koppelmerker im Block übertragen") notwendig, um zu verhindern, daß der Empfänger einen nur teilweise übertragenen Datensatz auswertet.

Lösung

Aufeinanderfolgende Datenwörter eines DB- oder DX- Datenbausteins, jeweils ab DW 0, werden als "Koppel-Datenwörter" definiert. Jede Verbindungsstrecke erhält "ihren" Datenbaustein und ist von den anderen Verbindungsstrecken völlig **un**abhängig.

Zu Beginn des Zyklus-Bausteins werden mit Hilfe der Sonderfunktions-Organisationsbausteine zur Mehrprozessor-Kommunikation die Koppel-Datenwörter empfangen. Es folgt das "reguläre" zyklische Programm, welches die empfangenen Daten auswertet und Sende-Daten erzeugt. Sie werden am Zyklusende wiederum mit Hife der Sonderfunktions-Organisationsbausteine zur Mehrprozessorkommunikation gesendet. So können sie von den anderen CPUs bei deren Zyklusbeginn empfangen werden.

Für jede der max. 12 möglichen Verbindungsstrecken und unabhängig von den anderen Verbindungsstrecken gilt:

- Die Sende-CPU wird nur aktiv, falls die Empfangs-CPU die "alten" Daten vollständig dem Zwischenspeicher KOR 923C entnommen hat.
- Die Empfangs-CPU wird nur aktiv, falls die Sende-CPU die "neuen" Daten vollständig im Zwischenspeicher KOR 923C abgelegt hat.

Somit steht der Empfangs-CPU entweder ein kompletter neuer Datensatz zur Verfügung oder der alte Datensatz bleibt unverändert: **Keine Mischung von "alten" und "neuen" Daten!**

Datenstruktur

Welche Datenwörter (nachfolgend Datenwortbereich genannt) von welcher CPU zu welcher CPU zu übertragen sind, ist in der Verbindungsliste (siehe Tabelle auf der nächsten Seite) beschrieben. Sie befindet sich in einem zusätzlichen Datenbaustein, welcher in allen beteiligten CPUs vorhanden sein muß.

Die Datenwortbereiche beginnen immer ab Datenwort DW 0, ihre Länge wird in Blöcken angegeben. Hierbei ist zu beachten:

- Ein kompletter Block besteht aus 32 Datenwörtern.
- Ist der letzte Block eines Sende-Datenbausteins "angeschnitten", d. h. umfaßt er zwischen einem und 31 Datenwörtern, so werden entsprechend weniger Datenwörter übertragen.
- Ist ein Sende-Datenbaustein länger als die in der Verbindungsliste angegebene Blockanzahl, so können die überzähligen Datenwörter in der entsprechenden CPU verwendet werden.
- Ist ein Empfangs-Datenbaustein länger als der empfangene Datenwortbereich, so können die überzähligen Datenwörter in der entsprechenden CPU verwendet werden.

Aufbau der Verbindungsliste

Tabelle 10-8 Verbindungsliste für die Erweiterung des Koppelmerkerbereichs

	TEILLISTE 1			**TEILLISTE 2**		
Verbindungs-strecke		DB-Typ	DB-Nummer			Block anzahl
von CPU 1 nach ...	DW 0	**S 1**		DW 16	**S 1**	
... CPU 2	DW 1	...	...	DW 17	**2**	...
... CPU 3	DW 2	...	...	DW 18	**3**	...
...CPU 4	DW 3	...	...	DW 19	**4**	...
von CPU 2 nach ...	DW 4	**S 2**		DW 20	**S 2**	
... CPU 1	DW 5	...	...	DW 21	**1**	...
... CPU 3	DW 6	1 [1)]	10 [1)]	DW 22	**3**	2 [1)]
... CPU 4	DW 7	...	...	DW 23	**4**	...
von CPU 3 nach ...	DW 8	**S 3**		DW 24	**S 3**	
... CPU 1	DW 9	...	...	DW 25	**1**	...
... CPU 2	DW 10	...	...	DW 26	**2**	...
... CPU 4	DW 11	...	...	DW 27	**4**	...
von CPU 4 nach ...	DW 12	**S 4**		DW 28	**S 4**	
... CPU 1	DW 13	...	...	DW 29	**1**	...
... CPU 2	DW 14	...	...	DW 30	**2**	...
... CPU 3	DW 15	...	...	DW 31	**3**	...
		2^{15}	2^{0}		2^{15}	2^{0}

[1)] Siehe Besipiel auf der nächsten Seite

Die Verbindungsliste besteht aus zwei ähnlich strukturierten Teillisten zu je 16 Datenwörtern. Ausgehend von jeder der vier Sende-CPUs (S1, S2, S3, S4) sind zur Beschreibung jeder Verbindungsstrecke 3 Einträge vorgesehen:

- **Blockanzahl**

  Die Anzahl der Blöcke legt die Größe (= Anzahl der Datenwörter) des zu übertragenden Datenwortbereiches fest. (Nicht vorhandene bzw. nicht genutzte Verbindungsstrecken werden mit Blockanzahl = 0 gekennzeichnet, ebenso bei DB-Typ und DB-Nummer.)

- **DB-Typ**

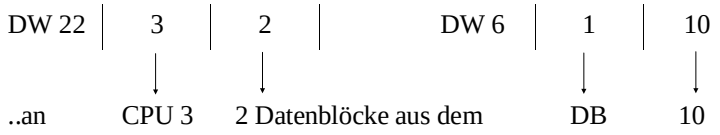
  Typ des Datenbausteins, der den auszusendenden Datenwortbereich enthält.

- **DB-Nummer**

  Nummer des Datenbausteins, der den auszusendenden Datenwortbereich enthält.

Diese Einträge können in der obigen Darstellung zeilenweise gelesen und ausgefüllt werden. Um beispielsweise die ersten zwei Datenblöcke aus dem Datenbaustein DB 10 von CPU 2 (S2) an die CPU 3 zu übertragen, ist folgender Eintrag notwendig:

CPU 2 (**S 2**) sendet ..



Die Teilliste 2 ist identisch mit der für die Funktion INITIALISIEREN (OB 200) benötigten Zuordnungsliste (Betriebsart "Manuell"). Innerhalb des Datenbausteins muß die Teilliste 1 die Datenwörter 0 bis 15 und die Teilliste 2 die Datenwörter 16 bis 31 belegen. Die mit Fettschrift hervorgehobenen Einträge dürfen nicht abgeändert werden.

Programmstruktur

Auf dem Koordinator werden von einer CPU beim Anlauf durch Aufruf der Funktion INITIALISIEREN (OB 200) genau so viele Speicherblöcke pro Verbindungsstrecke reserviert, wie Datenblöcke auf dieser Strecke zu übertragen sind.

Zum Aussenden und Empfangen der Datenwortbereiche werden auf jeder CPU zwei Funktionsbausteine verwendet:

FB-Nr.	Name	Funktion
FB 100	SEND-DAT	Senden von Datenwortbereichen an die restlichen CPUs
FB 101	EMPF-DAT	Empfangen von Datenwortbereichen von den restlichen CPUs

Die FB-Nummern sind zufällig gewählt und können geändert werden.

Die Funktionsbausteine SEND-DAT und EMPF-DAT entnehmen der Verbindungsliste, welche Datenwortbereiche aus welchen Datenbausteinen auszusenden oder zu empfangen sind. Es wird immer der **gesamte** Datenwortbereich ausgesendet oder empfangen. Falls dies, wegen unzureichender Sende- oder Empfangs-Kapazität, noch nicht möglich ist, wird auf das Senden bzw. Empfangen verzichtet.

Hinweis
Dieses Beispiel (Koppelmerkererweiterung mit Hilfe der Funktionsbausteine SEND-DAT und EMPF-DAT) kann nur dann korrekt ablaufen, wenn in keiner der CPUs zusätzlich die Sonderfunktions-Organisationsbausteine zur Mehrprozessorkommunikation OB 202 bis OB 205 aufgerufen werden.

Die Funktionsbausteine SEND-DAT und EMPF-DAT enthalten die Sonderfunktions-Organisationsbausteine zur Mehrprozessorkommunikation OB 202 bis OB 205. Der zusätzliche Aufruf dieser Organisationsbausteine außerhalb von SEND-DAT / EMPF-DAT ist nicht zulässig!

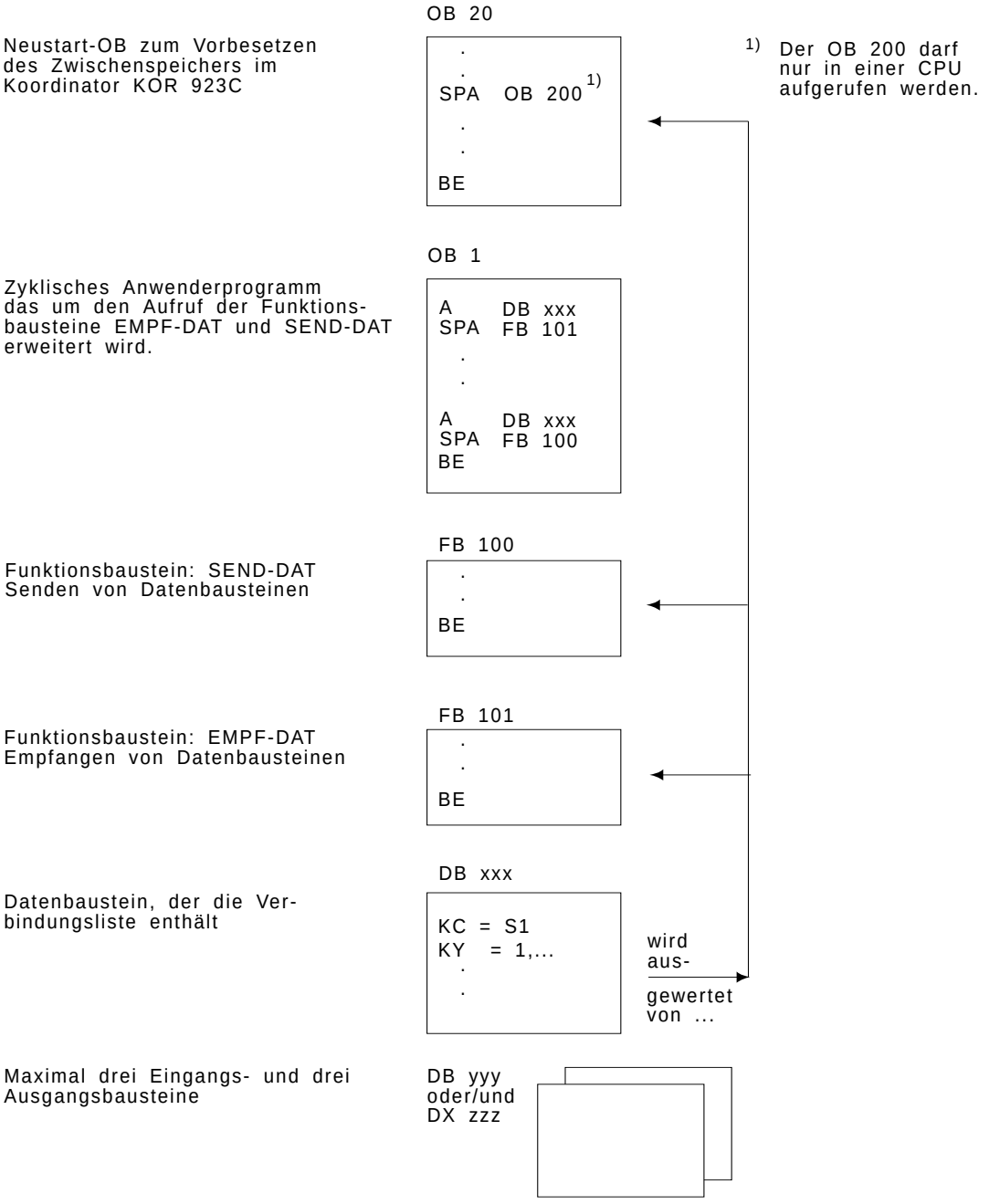
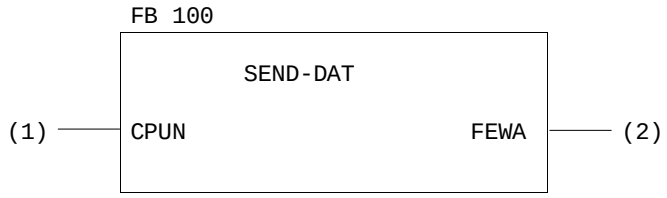


Bild 10-6 Übersicht über die benötigten Bausteine

Programmieren der Funktionsbausteine

FB 100: Senden von Datenwortbereichen

Vor Aufruf des FB 100 muß der Datenbaustein aufgeschlagen werden, der die Verbindungsliste enthält. Der Funktionsbaustein SEND-DAT benötigt zum Auswerten der in der Verbindungsliste enthaltenen Informationen die Nummer der CPU, auf der er aufgerufen wird.
Falls innerhalb des Funktionsbausteins die Funktion SENDEN (OB 202) nicht korrekt abgearbeitet werden konnte, wird die jeweilige Fehler- oder Warnungs-Nummer im Ausgangsparameter FEWA übergeben und das VKE = 1 gesetzt.
Zusätzlich enthält FEWA bei unzulässigem Eingangsparameter CPUN (CPU-Nummer) den Wert 16 (Bit-Nr. 4 = 1).
Der Funktionsbaustein SEND-DAT verwendet die Merkerbytes MB 239 bis MB 251 als Schmiermerker.



Parameter-Name	Bedeutung	Art	Typ
CPUN	**N**ummer der **CPU**, auf der der FB 100 aufgerufen wird. Zulässig sind die Nummern 1 bis 4.	D	KF
FEWA	**Fe**hler/**Wa**rnung (siehe Funktion SENDEN / OB 202)	A	BY

```
FB 100                                                    LAE=90

NETZWERK 1       0000
NAME:SEND-DAT
BEZ   :CPUN   E/A/D/B/T/Z:       D  KM/KH/KY/KC/KF/KT/KZ/KG: KF
BEZ   :FEWA   E/A/D/B/T/Z:       A  BI/BY/W/D:               BY

000B     :LW    =CPUN                     CPUN = CPUN - 1
000C     :L     KB 1                      Fehler falls:
000D     :-F
000E     :SPM   =FEWA                     CPU-Nr. <1
000F     :L     KB 3
0010     :>F
0011     :SPB   =FEWA                     CPU-Nr >4
0012     :TAK
```

Fortsetzung auf der nächsten Seite

Fortsetzung 1 von FB 100:

```
0013      :
0014      :SLW   2                          CPUN = CPUN * 4
0015      :T     MB 245                     Basisadresse
0016      :
0017      :L     KB 1
0018      :T     MB 244                     Verbindungszaehler
0019      :
001A SCHL :L     MB 245                     Basisadresse
001B      :L     MB 244                       + Zaehler
001C      :+F
001D      :T     MW 240
001E      :ADD   BF+16                        + Offset
001F      :T     MW 242
0020      :
0021      :B     MW 242
0022      :L     DR 0                       Anzahl der reservierten
0023      :T     MB 239                     Bloecke = 0 ?
0024      :L     KB 0
0025      :!=F
0026      :SPB   =LEER
0027      :
0028      :B     MW 242
0029      :L     DL 0                       Nr. der Empfangs-CPU
002A      :T     MB 246
002B      :L     KB 246                     SF-OB:
002C      :SPA OB 203                       "Sendemoeglichkeit testen"
002D      :L     MB 248                     Abbruch bei Fehler"
002E      :SPB   =OBFE
002F      :
0030      :L     MB 249                     Sende-Kapazitaet >< Anzahl
0031      :L     MB 239                     der reservierten Bloecke?
0032      :><F
0033      :SPB   =LEER
0034      :
0035      :L     KB 0                       Blockzaehler
0036      :T     MB 249
0037      :
0038      :B     MW 240
0039      :L     DW 0                       Typ und Nummer des
003A      :T     MW 247                     Quell-DB
003B      :
003C UEBT :L     KB 246                     SF-OB:
003D      :SPA   OB 202                     Senden eines Datenblocks
003E      :L     MB 250                     Abbruch bei Fehler/Warnung
003F      :SPB   =OBFE
0040      :
0041      :L     MB 249                     Block-Nr. = Block-Nr. + 1
0042      :I     1
0043      :T     MB 249                     alle Bloecke uebertragen ?
0044      :L     MB 239
0045      :<F
0046      :SPB   =UEBT
0047      :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 2 von FB 100:

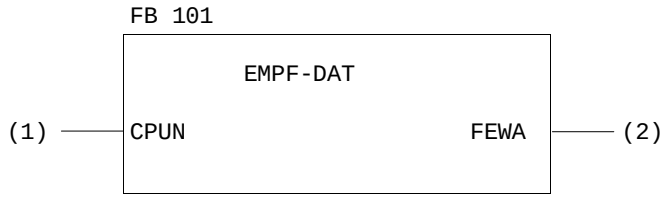
```
0048 LEER :L     MB 244                    Verbindungszaehler
0049      :I     1                         inkrementieren
004A      :T     MB 244
004B      :L     KB 4                      alle Verbindungs-
004C      :<F                              strecken bearbeitet ?
004D      :SPM   =SCHL
004E      :L     KB 0                      regulaeres Programmende:
004F      :T     =FEWA                     VKE = 0, FEWA = 0
0050      :BEA
0051      :
0052 FEWA :L     KB 16                     Programmende bei Fehler:
0053 OBFE :T     =FEWA                     VKE = 1, FEWA enthaelt
0054      :BE                              Fehler-/Warnungs-Nummer
```

FB 101: Empfangen von Datenwortbereichen

Vor Aufruf des FB 101 muß der Datenbaustein aufgeschlagen werden, der die Verbindungsliste enthält. Der Funktionsbaustein EMPF-DAT benötigt zum Auswerten der in der Verbindungsliste enthaltenen Informationen die Nummer der CPU, in der er aufgerufen wird.

Falls innerhalb des Funktionsbausteins die Funktion EMPFANGEN (OB 204) nicht korrekt abgearbeitet werden konnte, wird die jeweilige Fehler- oder WarnungsNummer im Ausgangsparameter FEWA übergeben und das VKE = 1 gesetzt. Zusätzlich enthält FEWA bei unzulässigem Eingangsparameter CPUN den Wert 16 (Bit-Nr. 4 = 1).

Der Funktionsbaustein EMPF-DAT verwendet die Merkerbytes MB 242 bis MB 255 als Schmiermerker.



Parameter-Name	Bedeutung	Art	Typ
CPUN	**N**ummer der **CPU**, auf der der FB 101 aufgerufen wird. Zulässig sind die Nummern 1 bis 4.	D	KF
FEWA	**Fe**hler/**Wa**rnung (siehe Funktion EMPFANGEN / OB 204)	A	BY

Fortsetzung auf der nächsten Seite

Fortsetzung 1 von FB 101:

```
FB 101                                                        LAE=88

NETZWERK 1        0000
NAME:EMPF-DAT
BEZ   :CPUN    E/A/D/B/T/Z:        D  KM/KH/KY/KC/KF/KT/KZ/KG: KF
BEZ   :FEWA    E/A/D/B/T/Z:        A  BI/BY/W/D:               BY

000B      :LW    =CPUN                      Fehler falls:
000C      :L     KB 1
000D      :<F
000E      :SPB   =FEWA                      CPU-Nr. <1
000F      :LW    =CPUN
0010      :L     KB 4
0011      :>F
0012      :SPB   =FEWA                      CPU-Nr >4
0013      :
0014      :L     KB 1                       Verbindungszaehler
0015      :T     MB 242
0016      :
0017      :L     KB 16
0018      :T     MW 244                     Zeiger auf Teilliste 2
0019      :
001A SUCH :L     MW 244                     Teilliste 2 solange durch-
001B      :I     1                          suchen, bis der naechste
001C      :T     MW 244                     Eintrag fuer die Empfangs-
001D      :B     MW 244                     CPU mit der Nummer'CPUN'
001E      :L     DL 0                       gefunden ist.
001F      :LW    =CPUN
0020      :><F
0021      :SPB   =SUCH
0022      :
0023      :B     MW 244
0024      :L     DR 0                       Anzahl der reservierten
0025      :T     MB 243                     Speicherbloecke = 0 ?
0026      :L     KB 0
0027      :!=F
0028      :SPB   =LEER
0029      :
002A      :L     MW 244                     Nummer der Sende-CPU
002B      :L     KM 00000000 00001100       aus dem Zeiger auf die
002D      :UW                               Teilliste 2 bestimmen.
002E      :SRW 2
002F      :I     1
0030      :T     MB 246
0031      :
0032      :L     KB 246                     SF-OB:
0033      :SPA   OB 205                     "Empfangsmoeglichkeit testen"
0034      :L     MB 248
0035      :SPB   = OBFE                     Abbruch bei Fehler
0036      :
```

Fortsetzung auf der nächsten Seite

Fortsetzung 2 von FB 101:

```
0037      :L     MB 249                    Empfangskapazitaet = Anzahl
0038      :L     MB 243                    der reservierten
0039      :><F                             Speicherbloecke ?
003A      :SPB   =LEER
003B      :
003C EMPF :L     KB 246                    SF-OB:
003D      :SPA   OB 204                    "Empfangen eines Datenblocks"
003E      :L     MB 248
003F      :SPM   =OBFE                     Abbruch bei Fehler/Warnung
0040      :L     MB 249                    bei Empfangskapazitaet = 0
0041      :L     KB 0                      naechste Verbindung-
0042      :><F                             strecke bearbeiten
0043      :SPB   =EMPF
0044      :
0045 LEER :L     MB 242                    Verbindungszaehler
0046      :I     1                         inkrementieren
0047      :T     MB 242
0048      :L     KB 4                      alle Verbindungsstrecken
0049      :<F                              bearbeitet ?
004A      :SPM=  SUCH
004B      :L     KB 0                      regulaeres Programmende:
004C      :T     =FEWA                     VKE = 0, FEWA = 0
004D      :BEA
004E      :
004F FEWA :L     KB 16                     Programmende bei Fehler:
0050 OBFE :T     =FEWA                     VKE = 1, FEWA enthaelt
0051      :BE                              Fehler-/Warnungs-Nummer
```

Anwendungsbeispiel

Anwendung der FB 100/101

Aufgabenstellung

Zwischen drei CPUs sollen Daten ausgetauscht werden:

- **von CPU 1 nach CPU 2:** Datenbaustein DB 3, DW 0 bis DW 127 (= 4 Blöcke)

- **von CPU 1 nach CPU 3:** Datenbaustein DX 4, DW 0 bis DW 63 (= 2 Blöcke)

- **von CPU 2 nach CPU 1 und CPU 3:** Datenbaustein DB 5, DW 0 bis DW 95 (= 3 Blöcke)

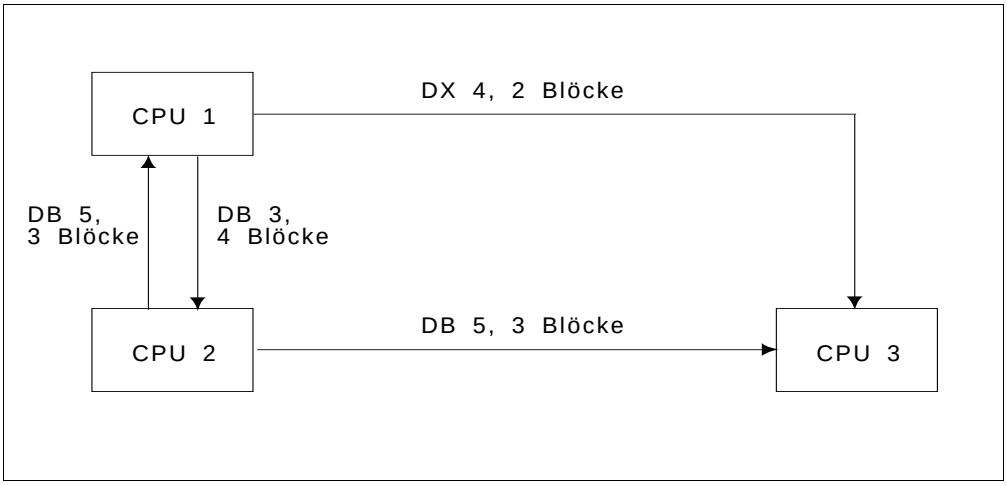


Bild 10-7 Datenaustausch zwischen 3 CPUs

Auf allen drei CPUs soll der Funktionsbaustein FB 1 die Schnittstelle zum zyklischen Anwenderprogramm bilden. Die CPU 1 soll bei NEUSTART die Funktion INITIALISIEREN (OB 200) aufrufen. Die Verbindungsliste soll im Datenbaustein DB 100 stehen.

Fortsetzung auf der nächsten Seite

Fortsetzung 1 des Anwendungsbeispiels:

Realisierung

1. Bausteine laden

Folgende Bausteine müssen zusätzlich zum OB 1 in die einzelnen CPUs geladen werden:

Funktion	CPU 1	CPU 2	CPU 3
Anlauf-OB	OB 20	—	—
Anwenderprogramm	FB 1	FB 1	FB 1
FB: SEND-DAT	FB 100	FB 100	FB 100
FB: EMPF-DAT	FB 101	FB 101	FB 101
Verbindungsliste	DB 100	DB 100	DB 100
Eingangs-DB	DB 5	DB 3	DB 5; DX 4
Ausgangs-DB	DB 3; DX 4	DB 5	—

2. Verbindungsliste erstellen

Die Verbindungsliste wird erstellt und in den Datenbaustein DB 100 eingetragen:

```
DB100                                                    LAE=37
                                                              BLATT    1

- - Teilliste 1 - -

 0:      KC = 'S1';                        von CPU 1 zu ..
 1:      KY = 001,003;                     .. CPU 2 den DB 3 senden
 2:      KY = 002,004;                     .. CPU 3 den DX 4 senden
 3:      KY = 000,000;
 4:      KC = S2;                          von CPU 2 zu ..
 5:      KY = 001,005;                     .. CPU 1 den DB 5 senden
 6:      KY = 001,005;                     .. CPU 3 den DB 5 senden
 7:      KY = 000,000;
 8:      KC = 'S3';
 9:      KY = 000,000;
10:      KY = 000,000;
11:      KY = 000,000;
12:      KC = 'S4';
13:      KY = 000,000;
14:      KY = 000,000;
15:      KY = 000,000;
```

Fortsetzung 2 des Anwendungsbeispiels:

```
– – Teilliste 1 – –

16:      KC = ’S1’;                        von CPU 1 zu ..
17:      KY = 002,004;                     .. CPU 2 vier Datenblöcke senden
18:      KY = 003,002;                     .. CPU 3 zwei Datenblöcke senden
19:      KY = 004,000;
20:      KC = S2’;                         von CPU 2 zu ..
21:      KY = 001,003;                     .. CPU 1 drei Datenblöcke senden
22:      KY = 003,003;                     .. CPU 3 drei Datenblöcke senden
23:      KY = 004,000;
24:      KC = ’S3’;
25:      KY = 001,000;
26:      KY = 002,000;
27:      KY = 004,000;
28:      KC = ’S4’;
29:      KY = 001,000;
30:      KY = 002,000;
31:      KY = 003,000;

Die Datenwörter DW 16 bis DW 31 enthalten die für die Funktion manuelles
INITIALISIEREN (OB 200) notwendige Zuordnungsliste.

3. Aufruf des OB 200 im Anlaufbaustein OB 20 für CPU 1 programmieren

Der OB 200 wird vom nachfolgend abgedruckten OB 20 der CPU 1 im Anlauf
aufgerufen.

OB 20                                                        LAE=yyABS

NETZWERK 1
0000     :L     KB 2                      manuelles Initialisieren der
0001     :T     MB 246                    Kacheln
0002     :
0003     :L     KY 1,100                  im DB 100 ist die Zuordnungsliste
0005     :T     MW 248                    ab dem Datenwort DW 16
0006     :L     KF+16                     eingetragen
0008     :T     MW 250
0009     :
000A     :L     KB 246                    SF-OB:
000B     :SPA   OB 200                    "Initialisieren"
000C     :
000D     :UN    M 252.5                   Bausteinende, wenn kein
000E     :BEB                             Initialisierungskonflikt
000F     :
0010     :                                hier wird die Fehlerbehandlung im
0011     :                                Falle eines Initialisierungs-
0012     :                                konfliktes eingefügt (z. B. Stopp,
0013     :                                Meldung auf Drucker ausgeben,
0014     :                                oder ...)

00xx     :BE
```

Fortsetzung auf der nächsten Seite

Fortsetzung 3 des Anwendungsbeispiels:

4. Aufrufe der Funktionsbausteine in den FB 1 der CPUs programmieren

Auf jeder CPU wird das Anwenderprogramm um den Aufruf der Funktionsbausteine EMPF-DAT und SEND-DAT erweitert. Der abgedruckte Funtionsbaustein FB 1 ist für die CPU 1 bestimmt. Für den Ablauf auf den anderen CPUs muß lediglich der Eingangsparameter CPUN (CPU-Nummer) angepaßt werden.

```
FB 1                                                        LAE=yy

NETZWERK 1       0000
NAME:EM-SE
0000
0000      :A     DB 100                    Verbindungsliste DB 100
0001      :SPA   FB 101                    Empfangen der Eingangs-
0002      :                                Datenbausteine
0003 NAME:EMPF-DAT
0004 CPUN:       KF+1
0005 FEWA:       MB 0
0006      :SPB   =FEWA                     Abbruch bei Fehler/Warnung
0007      :
0008      :
0009      :                                Hier wird das zyklische Anwender-
000A      :                                programm eingefuegt, das Daten aus
000B      :                                den Eingangsdatenbausteinen liest
000C      :                                und Daten in die Ausgangsdaten-
000D      :                                bausteine eintraegt.
000E      :
000F      :
0010      :A     DB 100                    Verbindungsliste DB 100
0011      :SPA   FB 100                         Senden der Ausgangsdaten-
0012      :                                bausteine
0012 NAME:SEND-DAT
0013 CPUN:       KF+1
0014 FEWA:       MB 0
0015      :SPB   =FEWA                     Abbruch bei Fehler/Warnung
0016      :BEA
0017      :
0018 FEWA:                                 nach Fehler/Warnung Fehlerbehandlung
0019      :                                durchfuehren (hier wird die Fehler-
001A      :                                behandlung eingefuegt z. B. Stopp,
001B      :                                Fehlermeldung auf Drucker oder
001C      :                                Monitor ausgeben, oder ...)

00xx      :BE
```

PG-Schnittstellen und -Testhilfen

11

Inhalt von Kapitel 11

PG-Schnittstellen und -Funktionen

11

Dieses Kapitel informiert Sie darüber, wie Sie Ihr PG an eine CPU 928B koppeln können und welche Hilfen Ihnen die PG-Software bietet, um Ihr STEP-5-Programm zu testen.
Wenn Sie nur die Standard-PG-Schnittstelle (1. serielle PG-Schnittstelle) benutzen, brauchen Sie den Abschnitt 11.4 nicht zu lesen.
Diese Abschnitte informieren Sie darüber, wie Sie über weitere Schnittstellen ein PG mit Ihrer CPU koppeln können. Sie erfahren darin ferner, was Sie bei der Benutzung der PG-Funktionen auf beiden Schnittstellen beachten müssen.

11.1 Übersicht

Das Laden und Testen Ihres Anwenderprogramms führen Sie mit Hilfe der Online-Funktionen der STEP-5-Software durch.

Dazu müssen Sie Ihre CPU mit dem PG koppeln. Für diese Kopplung stehen Ihnen folgende Schnittstellen zur Verfügung:

- Kopplung über die serielle Standard-Schnittstelle "PG - AG",
- Kopplung über die 2. serielle Schnittstelle der CPU 928B.

Die PG-Funktionen laufen auf den beiden seriellen Schnittstellen parallel ab, über SINEC-H1-Kopplung dagegen die PG-Funktionen nur **alternativ** zu denen auf den seriellen Schnittstellen.

Die PG-Funktionen bieten Ihnen folgende Unterstützung für die Inbetriebnahme und den Test Ihres STEP-5-Programms :

Tabelle 11-1 Hilfen für Inbetriebnahme und Test

Hilfe	Abschnitt
Auskunft	
höchste nutzbare Adresse des RAM-Modus und letzte belegte Adresse des Speichermoduls	"Speicherausbau"
Liste der geladenen Bausteine	"Bausteinliste"
Inhalt von Speicherwörtern/bytes und Peripheriebytes anzeigen	"Ausgabe Adresse"
Speicherverwaltung	
Speicher komplett löschen	"Urlöschen"
freien Speicherplatz zusammenfassen	"Speicher komprimieren"
Bausteine verwalten	"Baustein übertragen/löschen"
Programmtest	
CPU starten/anhalten	"Start/Stopp"
Befehlsfolge in einem Baustein testen	"Status"
einzelne Programmschritte testen	"Bearbeitungskontrolle"
Signalzustand von Prozeßvariablen anzeigen	"Status Variablen"
Ausgangssignale im Stopp ausgeben	"Steuern"
Prozeßvariable anzeigen/verändern	"Steuern Variablen"

11.2 PG-Funktionen

Hinweis
Die in diesem Abschnitt für die PG-Funktionen verwendeten Begriffe können u. U. von den Begriffen in Ihrer PG-Software abweichen.
Lesen Sie dazu bitte in Ihrem STEP-5-Handbuch nach.

Aufrufen und Bedienen

Wie Sie die einzelnen PG-Funktionen aufrufen und bedienen, lesen Sie bitte im Handbuch zu Ihrem Programmiergerät nach.

Ausführung

Im Automatisierungsgerät werden die PG-Funktionen an definierten Punkten ausgeführt. Hierbei gibt es Punkte im Systemprogramm (= Systemkontrollpunkte) und Punkte im Anwenderprogramm (= Anwenderkontrollpunkte).

Systemkontrollpunkte

Im Betriebszustand STOP existiert der Systemkontrollpunkt "**Stop**", der regelmäßig aufgerufen wird.

Im Betriebszustand RUN wird der Systemkontrollpunkt "**Zyklus**" am Ende der Programmbearbeitungsebene ZYKLUS vor der Prozeßabbildaktualisierung aufgerufen.

Befindet sich die CPU im WARTEZUSTAND, so wird dort regelmäßig der Systemkontrollpunkt **"Wartezustand"** aufgerufen.

Zusätzlich gibt es einen zeitbedingten Systemkontrollpunkt "**Asynchron**". Dieser Systemkontrollpunkt wird asynchron während der Programmbearbeitung eingeschachtelt.

Anwenderkontrollpunkte

Bei den Testfunktionen "Status Baustein" und "Bearbeitungskontrolle" werden Anwenderkontrollpunkte verwendet. Ein Anwenderkontrollpunkt wird aufgerufen, wenn ein Befehl ausgeführt ist, der vom PG markiert ist.

Betriebszustand WARTEZUSTAND

Bisher sind Ihnen die Betriebszustände STOP, ANLAUF und RUN bekannt. Bei der PG-Funktion "Bearbeitungskontrolle" nimmt die CPU einen weiteren Betriebszustand ein: den WARTEZUSTAND.
Wenn die CPU sich im WARTEZUSTAND befindet, können noch weitere PG-Funktionen aufgerufen werden.

Eigenschaften

- Im Wartezustand findet keine Bearbeitung des Anwenderprogramms statt.
- LEDs auf der Frontplatte:

  RUN-LED: aus
  STOP-LED: aus
  BASP-LED: an

- Alle Zeitzellen sind "eingefroren", d. h. es laufen keine Timer (die Zeitzellen werden nicht verändert). Ebenso bleiben alle Systemzeiten stehen, wie die der Regelung und die der zeitgesteuerten Bearbeitung.
  Nach Verlassen des WARTEZUSTANDs laufen die Timer weiter.
- Unterbrechungsursachen wie z. B. PEU, BAU, MPSTP oder Stoppschalter werden im WARTEZUSTAND registriert, aber es erfolgt keine Reaktion darauf.

Unterbrechungen

Wenn im WARTEZUSTAND Unterbrechungsursachen registriert worden sind, so werden die dazugehörigen Programmbearbeitungsebenen unmittelbar nach Verlassen des WARTEZUSTANDs aufgerufen.

Tritt NAU auf, so wird der WARTEZUSTAND verlassen und die Online-Funktion BEARBEITUNGSKONTROLLE abgebrochen. Nach NETZ EIN ist in den Steuerbits BARBEND angekreuzt. Der Stoppzustand kann nur mit NEUSTART verlassen werden.

11.2.1 Auskunft

Speicherausbau

Die PG-Funktion "Speicherausbau" zeigt Ihnen am PG die höchste nutzbare Adresse des RAM-Moduls an (bei EPROM wird '0' angezeigt) und die letzte mit Bausteinen des Anwenderprogramms belegte Adresse des Speichermoduls.

Ausgabe Adresse

Mit der Funktion "Ausgabe Adresse" können Sie sich am PG den Inhalt von Speicher- und Peripherieadressen hexadezimal anzeigen lassen. Sie können alle Adressen ansprechen. Im Bereich des Prozeßabbildes wird kein ADF ausgelöst,wenn zum entsprechenden Byte keine Peripheriebaugruppe vorhanden ist; im Peripheriebereich entsteht kein QVZ.

Im byteadressierbaren Bereich (Merker, Prozeßabbild) wird das High-Byte als 'FF' dargestellt.

Im Peripheriebereich wird bei quittierenden Adressen das Highbyte als '00' ausgegeben. Quittiert eine Peripherieadresse nicht, wird das Highbyte als 'FF' angezeigt.

11.2.2 Speicher- und Übertragungsfunktionen

Urlöschen

Sie können mit der Funktion "Löschen alle Bausteine" eine CPU vom PG aus urlöschen. Dabei wird das Urlöschen unbedingt durchgeführt (siehe Abschnitt 4.3.2).

Befindet sich die CPU beim Aufruf von "Löschen alle Bausteine" im Zustand ANLAUF oder RUN, wird zunächst ein Übergang in den Stoppzustand durchgeführt. Dabei wird – wenn geladen – der Organisationsbaustein OB 28 aufgerufen.

> **Hinweis**
> "Urlöschen" ist nicht zulässig, solange "Bearbeitungskontrolle" aktiv ist!

Speicher komprimieren

Diese Funktion optimiert die Speicherbelegung durch Bausteine: Der Platz, den ungültig markierte Bausteine belegen, wird so durch die vorhandenen gültigen Bausteine des Anwenderprogramms überschrieben (der Baustein wird auf einen anderen Speicherplatz umgeladen), daß diese von Anfang des Speichers hintereinanderliegen.
Dies wird getrennt im RAM-Modul und im DB-RAM durchgeführt und kann an den Systemkontrollpunkten "Zyklus" und "Stop" erfolgen.

Bei der CPU 928B ist die Funktion "Speicher komprimieren" im STOP immer möglich, auch wenn der BSTACK nicht leer ist.

Vorsicht
Nach "Speicher komprimieren" im STOP ist als Anlaufart nur NEUSTART zulässig. USTACK und BSTACK werden nicht nachgeführt.

Netzspannungsausfall beim Komprimieren

Fällt während des Komprimierens die Netzspannung aus, so wird nach Netzwiederkehr der Vorgang abgeschlossen und kein weiterer Baustein umgeladen. Bei erneutem Aufruf von "Speicher komprimieren" nach der Optimierungsvorgang fortgesetzt.

Fehler im Bausteinspeicher

Die Funktion "Speicher komprimieren" erkennt folgende Fehler im Bausteinspeicher:

- falsche Bausteinlänge,
- verfälschtes Muster '7070' im Bausteinkopf,
- ungültiger Bausteintyp (bei OBs ungültige Bausteinnummer).

Die Funktion wird daraufhin abgebrochen und am PG eine Meldung ausgegeben. Sie müssen dann zuerst "Urlöschen"; darauf ist der Speicher leer.

Hinweis
"Speicher komprimieren" ist nicht zulässig, solange "Bearbeitungskontrolle" aktiv ist!

Baustein übertragen

Mit dieser Funktion übertragen Sie neue oder schon vorhandene Code- und Datenbausteine in den Anwenderspeicher bzw. in das interne DB-RAM der CPU.

Ist ein Baustein bereits im Anwenderspeicher bzw. im internen DB-RAM der CPU vorhanden, so wird er als ungültig markiert und der neue erhält die Gültigmarkierung. Ein Baustein wird nur dann für ungültig erklärt, wenn er momentan nicht bearbeitet wird.

Baustein löschen

Mit dieser Funktion erkären Sie Code- und Datenbausteine im Anwenderspeicher der CPU für ungültig. Ein Baustein wird nur dann für ungültig erklärt, wenn er momentan nicht bearbeitet wird.

Den Platz, an dem diese Bausteine gelegen haben, können Sie über die Funktion "Speicher komprimieren" für andere Bausteine ausnutzen.

11.2.3 Programmtest

Start/Stopp

Die PG-Bedienung entspricht der manuellen Bedienung.

Mit dem Aufruf der PG-Funktion "Stopp" im Betriebszustand RUN bringen Sie das Automatisierungsgerät in den Stoppzustand.

Bei derjenigen CPU, an der das PG angeschlossen ist, sehen Sie folgendes Bild:

STOP-LED: an

BASP-LED: an

In den Steuerbits ist PG-STP angekreuzt. Bei Mehrprozessorbetrieb ist bei den anderen CPUs das Steuerbit MP-STP gesetzt.

Sie können eine CPU mit NEUSTART oder WIEDERANLAUF starten. Im Einzelprozessorbetrieb verläßt die CPU den Stoppzustand. Im Mehrprozessorbetrieb wird zunächst die Anlaufart voreingestellt (Steuerbit NEUST oder M W A ist gesetzt), die CPU bleibt aber im STOP. Mit der nachfolgenden Bedienung "System starten" können Sie das Automatisierungsgerät starten. Dies entspricht der Bedienung über den Koordinator (Schalter auf RUN).

Mit der PG-Funktion "Start" können Sie im Mehrprozessorbetrieb nacheinander bei allen CPUs den gewünschten Anlauf durchführen und erst bei der letzten CPU das AG starten.

- PG-Funktion NEUSTART:
  Es wird ein MANUELLER NEUSTART der CPU durchgeführt.

- PG-Funktion WIEDERANLAUF:
  Abhängig von der Einstellung im DX 0 wird ein MANUELLER WIEDERANLAUF oder ein MANUELLER NEUSTART MIT GEDÄCHTNIS durchgeführt.

Status Baustein

Mit Hilfe der Funktion "Status" testen Sie an beliebiger Stelle im Anwenderprogramm zusammenhängende Befehlsfolgen (STEP-5-Operationen) in einem Baustein.
Zu jedem ausgeführten Befehl im Baustein werden die aktuellen Signalzustände der Operanden, die AKKU-Inhalte, das VKE usw. am Programmiergerät ausgegeben (Schrittbetrieb). Auch die Parametrierung von Funktionsbausteinen kann auf diese Weise getestet werden : Angezeigt werden die Signalzustände der Aktualoperanden.

Funktion aufrufen und Haltepunkt vorgeben

Wenn Sie die Funktion "Status" am PG aufrufen und Bausteinart und Bausteinnummer (evtl. mit Schachtelreihenfolge und Suchbegriff) des zu testenden Bausteins eingeben, so geben Sie damit einen sog. "Haltepunkt" vor.

Bei Aufruf der Funktion während der Programmbearbeitung im RUN wird die Programmbearbeitung solange fortgesetzt, bis der durch den vorgegebenen Haltepunkt markierte Befehl in der richtigen Schachtelfolge erreicht ist. Danach werden die überwachten Befehle jeweils bis zur Befehlssgrenze ausgeführt und die Ergebnisse der Befehlsbearbeitung am PG ausgegeben.

Aufruf im STOP

Die Funktion "Status" läßt sich auch im STOP aufrufen. Danach ist sowohl ein NEUSTART als auch ein MANUELLER WIEDERANLAUF möglich. Die CPU bearbeitet daraufhin das Anwenderprogramm bis zum vorgegebenen Haltepunkt. Dann werden die Daten zu der gewünschten Befehlsfolge ausgegeben. Somit eignet sich die Funktion "Status" zum Beispiel auch dazu, das Anwenderprogramm im Anlauf oder im ersten Zyklus zu testen.

> **Hinweis**
> Die Ergebnisse der Befehlsbearbeitung werden nicht in jedem Programmzyklus ausgegeben.

Einschachtelungen und Unterbrechungen

Eine durch einen vorgegebenen Haltepunkt markierte Befehlsfolge wird vollständig durchlaufen, auch wenn zwischendurch an einer Befehlsgrenze eine andere Programmbearbeitungsebene (z. B. ein Fehler-OB, ein Prozeß- oder ein Weckalarm) eingeschachtelt und abgearbeitet wird.
Führt in einer eingeschachtelten Programmbearbeitungsebene eine Unterbrechungsursache die CPU in den Stoppzustand, so werden im STOP die Daten bis zu demjenigen Befehl ausgegeben, der als letzter vor der Einschachtelung ausgeführt worden ist. Die Daten der restlichen Befehle werden mit '0' aufgefüllt (auch SAZ = 0).

Wenn die CPU von einem Betriebszustand in den anderen wechselt (z. B. RUN → STOP → MANUELLER WIEDERANLAUF), bleibt die Funktion weiterhin aufgerufen. Beendet wird "Status" durch Betätigen der Abbruchtaste am Programmiergerät.

Bearbeitungskontrolle

Mit der Funktion "Bearbeitungskontrolle" testen Sie an beliebiger Stelle im Anwenderprogramm einzelne Programmschritte. Dazu halten Sie die Programmbearbeitung an und lassen die CPU dann einen Befehl nach dem anderen bearbeiten. Zu jedem ausgeführten Befehl werden die aktuellen Signalzustände von Operanden, die AKKU-Inhalte, das VKE usw. am Programmiergerät ausgegeben.

Funktion aufrufen und 1. Haltepunkt vorgeben

Zum Aufruf der Funktion "Bearbeitungskontrolle" geben Sie Bausteinart und Bausteinnummer (eventuell mit Schachtelreihenfolge) des zu testenden Bausteins an und markieren am PG den ersten Befehl, dessen Daten ausgegeben werden sollen. Damit geben Sie einen ersten Haltepunkt vor.
In den Steuerbits wird BARB angekreuzt. Die Befehlsausgabe wird gesperrt (BASP-LED = an).

Vorsicht
Wenn Sie am **Koordinator Testbetrieb** einstellen, wird die Befehlsausgabe **nicht** gesperrt (BASP-Led = aus). Werden jetzt Befehle bearbeitet, die die digitale Peripherie ändern, oder führt die CPU die Prozeßabbildaktualisierung durch, geben die Signalformer entsprechende Signale aus.

Aufruf im ANLAUF und im RUN

Wenn Sie den 1. Haltepunkt während der **Programmbearbeitung** im ANLAUF oder RUN vorgeben, setzt die CPU die Programmbearbeitung solange fort, bis der durch den vorgebenen Haltepunkt markierte Befehl erreicht ist. Der Befehl wird noch komplett ausgeführt. (Die Befehle BMW und BDW werden **einschließlich** substituiertem Befehl bearbeitet.)
Anschließend geht die CPU in den WARTEZUSTAND. Dort werden die Daten des markierten und zuletzt bearbeiteten Befehls ausgegeben.

Aufruf im STOP

Auch im STOP können Sie die Funktion BEARBEITUNGSKONTROLLE aufrufen und einen ersten Haltepunkt vorgeben. Die CPU bleibt weiterhin im Stoppzustand. Sie können jetzt sowohl einen NEUSTART als auch einen MANUELLEN WIEDERANLAUF durchführen. Die CPU führt die Programmbearbeitung bis zum markierten Befehl aus und verfährt dann wie oben.

Funktion fortführen und weiteren Haltepunkt vorgeben

Ausgangspunkt: Die CPU befindet sich im WARTEZUSTAND.

Um die Funktion fortzuführen, haben Sie zwei Möglichkeiten.

1. Sie geben einen **Folge**haltepunkt vor:

   Der vorgegebene Haltepunkt wird um einen Befehl verschoben. Die CPU verlässt den WARTEZUSTAND und setzt die Programmbearbeitung um diesen einen Befehl fort. Wenn der Befehl komplett bearbeitet ist, geht die CPU erneut in den WARTE-ZUSTAND und gibt dort die Daten aus.

   Wird der Folgebefehl jedoch in einer eingeschachtelten Programmbearbeitungsebene erreicht, setzt die CPU die Programmbearbeitung fort. Der Folgehaltepunkt bleibt weiterhin vorgegeben.

Hinweis
Im Stoppzustand können Sie **keinen** Folgehaltepunkt vorgeben!

2. Sie geben einen **neuen** Haltepunkt vor:

   Sie geben am PG einen beliebigen anderen Befehl im gleichen oder in einem anderen Baustein vor. Die CPU setzt die Programmbearbeitung fort, bis sie den neuen Haltepunkt erreicht. Der Befehl wird vollständig bearbeitet. Dann geht die CPU in den WARTE-ZUSTAND und gibt dort die Daten aus.

   Sie können die CPU mit Bearbeitungskontrolle auch um einen ganzen Zyklus weiterlaufen lassen (zyklusweise testen). Dafür setzen Sie im WARTEZUSTAND den Haltepunkt auf denselben Befehl, wie vorher. Allerdings darf sich der Befehl nicht in einer Programmschleife befinden. In diesem Fall wird die Schleife einmal durchlaufen; es erfolgt keine Programmbearbeitung über die Zyklusgrenze hinweg.

Hinweis
Im WARTEZUSTAND können Sie andere Funktionen wie "Ausgabe Buch", "Status Variablen" oder "Steuern Varaiable" aufrufen. Sobald die Programmbearbeitung nach Verlassen des WARTEZUSTANDs fortgesetzt wird, laufen die Timer und die Systemzeiten weiter, bis wieder ein Haltepunkt erreicht wird.

Haltepunkt zurücknehmen

Ist ein vorgegebener Haltepunkt noch nicht erreicht, so haben Sie die Möglichkeit, diesen nachträglich zurückzunehmen, indem Sie am PG die Abbruchtaste betätigen. Die CPU geht daraufhin in den WARTE-ZUSTAND. Danach können Sie einen neuen Haltepunkt vorgeben oder "Bearbeitungskontrolle Ende" aufrufen.

Funktion abbrechen

Durch Aufrufen von "Bearbeitungskontrolle Ende" können Sie während der Programmbearbeitung im WARETEZUSTAND und im STOP die Funktion abbrechen. Die CPU geht in den STOP (bzw. bleibt im STOP). Die STOP-LED blinkt langsam. In den Steuerbits wird BARBEND angekreuzt. Anschließend ist ein NEUSTART erforderlich.
Tritt während der Funktion "Bearbeitungskontrolle" ein Schnitt-stellenfehler auf (Unterbrechung am PG-Kabel), wird die Funktion abgebrochen wie oben beschrieben.

Einschachtelungen

Bei aufgerufener Funktion "Bearbeitungskontrolle" können nach Verlassen des WARTEZUSTANDs andere Programmbearbeitungsebenen eingeschachtelt werden.

Wenn der Befehl am Haltepunkt bearbeitet ist und an dieser Stelle eine andere Programmbearbeitungsebene aufgerufen ist (z. B. ein Fehler-OB oder ein Alarm-OB), so wird diese erst eingeschachtelt und vollständig bearbeitet, wenn der WARTEZUSTAND wieder verlassen wird.

Hinweis
Die Daten werden an der Befehlsgrenze gelesen und dort ausgegeben. Alle dann evtl. folgenden Einschachtelungen sind noch nicht bearbeitet.

Das Ablaufprinzip der Funktion "Bearbeitungskontrolle" zeigt Bild 11-1.

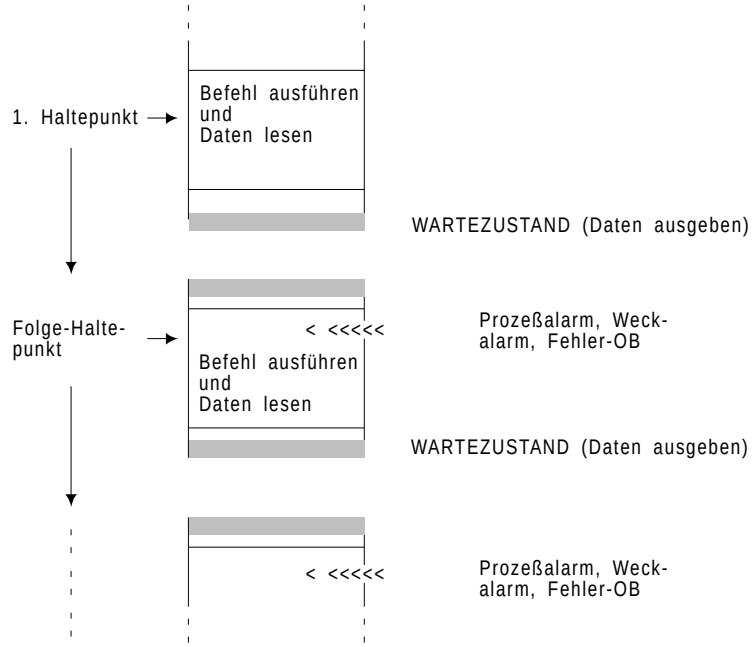


Bild 11-1 Testablauf bei "Bearbeitungskontrolle"

Sind im WARTEZUSTAND Anforderungen wie PEU, MP-STP, Stoppschalter usw. aufgetreten, werden diese nur registriert. Sofort nach Verlassen des WARTEZUSTANDs können diese wirksam werden: Eine Programmbearbeitungsebene wird eingeschachtelt oder eine Unterbrechung führt in den STOP.
Es gilt die Reihenfolge der Ereignisse. Gleichzeitige Anforderungen werden prioritiert.

Hinweis
Wenn die CPU im WARTEZUSTAND ist und eine Einschachtelung angefordert ist, haben Sie die Möglichkeit, einen Haltepunkt auf einen Befehl der Einschachtelung zu setzen. So können Sie z. B. bei einem Befehl, der einen QVZ auslöst, direkt danach den QVZ-Fehler-OB beobachten.

Unterbrechungen

- Programmbearbeitung (ANLAUF/RUN) → Stoppzustand:

  Tritt während der Programmbearbeitung eine Unterbrechung auf (z. B. Mehrprozessor-Stopp, Peripherie unklar, STOP-Schalter, Fehler-OB nicht programmiert usw.) und der vorgegebene Haltepunkt ist noch nicht erreicht, so geht die CPU in den Stoppzustand. Wenn danach ein Anlauf (NEUSTART oder MANUELLER WIEDERANLAUF) durchgeführt wird, bleibt die Funktion "Bearbeitungskontrolle" weiterhin aufgerufen; der Haltepunkt ist weiterhin vorgegeben.

- Befehlsbearbeitung am Haltepunkt (ANLAUF/RUN) → Stoppzustand:

  Wenn während der Befehlsbearbeitung am Haltepunkt oder Folgehaltepunkt Stopp-Bedingungen auftreten (STOP-Schalter, STEP-5-Befehl STP, Fehler-OB nicht programmiert), so geht die CPU unmittelbar nach der Befehlsbearbeitung in den STOP.
  Wird im Stoppzustand kein neuer Haltepunkt vorgegeben, geht die CPU nach einem ANLAUF in den WARTEZUSTAND. Die Funktion "Bearbeitungskontrolle" bleibt weiterhin aufgerufen.

- Wartezustand → STOP

  Unterbrechungsursachen, die im WARTEZUSTAND auftreten (z. B. z. B. Mehrprozessor-Stopp, Peripherie unklar, STOP-Schalter) oder vom vorausgegangenen Befehl stammen (Fehler, der in STOP führt), werden zwar registriert, die CPU bleibt jedoch im WARTEZUSTAND. Erst wenn im WARTEZUSTAND ein neuer Haltepunkt vorgegeben wird und die CPU den WARTEZUSTAND verläßt, bewirken die aufgetretenen Unterbrechungsursachen einen Übergang in den STOP. Der vorgegebene Haltepunkt wird nicht erreicht.
  Wird jetzt ein ANLAUF (NEUSTART oder MANUELLER WIEDERANLAUF) durchgeführt, ist der neue Haltepunkt weiterhin vorgegeben.

Hinweis
Wird im WARTEZUSTAND der Betriebsartenschalter auf STOP gestellt, so geht die CPU erst nach Verlassen des WARTEZUSTANDs in den STOP.

Führen Unterbrechungsursachen die CPU während der "Bearbeitungskontrolle" in den STOP, bleibt nach einem anschließenden ANLAUF die Funktion "Bearbeitungskontrolle" (und ein evtl. vorgegebener Haltepunkt) weiterhin aktiv!

Status Variablen

Mit der PG-Funktion "Status Variablen" können Sie sich die aktuellen Signalzustände bestimmter Operanden (Prozeßvariablen) anzeigen lassen.

Die Funktion aktiviert Systemkontrollpunkte im ZYKLUS, im STOP und im WARTEZUSTAND.

Wenn ein Kontrollpunkt erreicht ist, wird der zu diesem Zeitpunkt aktuelle Signalzustand der gewünschten Prozeßvariablen ausgegeben. Sie können alle Prozeßvariablen angeben (Eingänge, Ausgänge, Merker, Zeitzellen, Zähler und Datenwörter). Im Bereich des Prozeßabbilds wird bei Zugriff auf eine Adresse, zu der keine Peripherie vorhanden ist, kein ADF ausgelöst.

*Ablauf während der **Programmbearbeitung***

Läuft die Funktion im Betriebszustand RUN oder ANLAUF, so wird die Programmbearbeitung fortgesetzt, bis der Systemkontrollpunkt "Zyklus" erreicht ist. Dann werden die Signalzustände der Operanden am Zyklusende abgefragt und angezeigt. Eingänge werden aus dem **Prozeßabbild** gelesen. Solange die Funktion nicht abgebrochen wird, werden bei laufender Programmbearbeitung die Signalzustände zyklisch aktualisiert. Die Signalzustände werden dabei nicht an jedem Systemkontrollpunkt abgefragt.
Wird der Systemkontrollpunkt "Zyklus" nicht erreicht, erfolgt keine Ausgabe der Signalzustände (z. B. bei einer Dauerschleife im Anwenderprogramm)!

*Ablauf der Funktion im **STOP***

Wenn die Funktion "Status Variablen" im STOP läuft, werden die Signalzustände der Operanden ausgegeben, wie sie am Systemkontrollpunkt "Stop" vorliegen. Wichtig ist dabei, daß die **Eingänge direkt** von der Peripheriebaugruppe abgefragt und ausgegeben werden. Dadurch läßt sich zum Beispiel testen, ob ein Peripherie-Eingangssignal tatsächlich zur CPU gelangt. Sie können auch im Mehrprozessorbetrieb alle Eingänge angeben, unabhängig von der Zuteilung im DB 1. Die Ausgänge werden vom Prozeßabbild gelesen.

*Ablauf der Funktion im **WARTEZUSTAND***

Die Funktion "Status Variablen" können Sie auch aufrufen, wenn die CPU sich mit der Funktion "Bearbeitungskontrolle" im WARTEZUSTAND befindet. Am Systemkontrollpunkt "Wartezustand" werden die Signalzustände der Operanden abgefragt und ausgegeben. Wie im Stoppzustand werden dabei die Eingänge direkt, die Ausgänge dagegen aus dem **Prozeßabbild** gelesen.

Wechsel des Betriebszustandes/ Beenden der Funktion

Wenn die CPU von einem Betriebszustand in den anderen wechselt (z. B. RUN → STOP → MANUELLER WIEDERANLAUF), bleibt die Funktion weiterhin aufgerufen. Beendet wird "Status Variablen" durch Betätigen der Abbruchtaste am Programmiergerät.

> **Hinweis**
> Die Variablen werden nicht in jedem Programmzyklus ausgegeben.

Steuern

Mit Hilfe der Funktion "Steuern" können Sie die Ausgangsbytes des Automatisierungsgerätes direkt (unter Umgehung des Prozeßabbildes) auf einen gewünschten Signalzustand einstellen oder nicht quittierende Signalformer (digitale Peripherie 0 bis 127) erkennen (Meldung am PG). Sie haben die Möglichkeit, die von den Ausgängen versorgten Prozeßgeräte ("Aktoren" wie z. B. Motor, Ventil) direkt zu überprüfen und zu steuern.

Hinweis
Die Funktion "Steuern" ist nur im **STOP** zulässig!

Ablauf der Funktion

Bei Aufruf der Funktion im STOP wird die Befehlsausgabesperre aufgehoben (BASP = inaktiv). Die **gesamte** digitale Peripherie (F000H bis F07FH) wird gelöscht, wobei jede Adresse mit dem Wert '0' beschrieben wird. Während des Löschens der Peripherie ist die Funktion nicht unterbrechbar.
Die Peripherieausgänge werden byteweise gesteuert, direkt und ohne das Prozeßabbild der Ausgänge zu beeinflussen!
Im Mehrprozessorbetrieb können Sie **alle** Peripherieausgänge steuern (unabhängig von einer Peripheriezuteilung im DB 1).

Wenn die Funktion aktiv ist (Meldung "Steuern fertig" am PG), können Sie einen NEUSTART oder einen MANUELLEN WIEDERANLAUF durchführen. Nach einem erneuten Übergang in den STOP können Sie wieder steuern. Die Ausgangssignalformer werden in diesem Fall **nicht gelöscht.**

Beenden der Funktion

Sie beenden die Funktion durch Drücken der Abbruchtaste am PG. Die Befehlsausgabesperre ist wieder aktiv (Leuchtdiode BASP = an).

Steuern Variablen

Mit der PG-Funktion "Steuern Variablen" können Sie die Werte von Operanden (Prozeßvariablen) einmalig verändern. Dies ist in jedem Betriebszustand der CPU zulässig. Sie können alle Prozeßvariablen angeben. Im Bereich des Prozeßabbilds wird bei Zugriff auf eine Adresse, zu der keine Peripherie vorhanden ist, kein ADF ausgelöst.
Die Änderung wird an den Systemkontrollpunkten "asynchron", d. h. erst am Zyklusende, wirksam. Beachten Sie, daß die gesteuerten Werte nachträglich überschrieben werden können (z. B. durch das Anwenderprogramm oder die Prozeßabbildaktualisierung)!

Hinweis
Das PG steuert die Prozeßvariablen E, A, M byteweise und DW, T, Z wortweise.

Wenn Sie **mehrere Operanden** steuern, dann werden die geänderten Bytes (bei DW, T Z die Wörter) nacheinander, über mehrere Funktionsaufrufe verteilt, im Speicher geändert.

11.3 Tätigkeiten an Kontrollpunkten

Der nchfolgenden Tabelle können Sie entnehmen, welche Tätigkeiten bei den PG-Funktionen an den Kontrollpunkten ausgeführt werden.

Tabelle 11-2 Tätigkeiten an Kontrollpunkten

Tätigkeiten der Online-Funktionen	Sytemkontrollpunkt				Anwender-kontroll-punkt
	"Stop"	"Zyklus"	"Warte-zustand"	"Asyn-chron"	
Eingabe der Adresse: Daten schreiben [1]	*		*	*	
Bausteineingabe: Baustein gültig erklären	*	*	*	*	
Baustein löschen	*		*	*	
Speicher komprimieren: Baustein verschieben [1] [2] [3]	*	*			
START/STOP	*	*	*	*	
URLÖSCHEN	*				
STATUS: Daten lesen und ausgeben					*
STATUS VARIABLEN: Daten lesen und ausgeben	*	*	*		
BEARBEITUNGSKONTROLLE: Haltepunktvorgabe	*	*	*	*	
BEARBEITUNGSKONTROLLE: Daten lesen und ausgeben					*
STEUERN (Signalformer) [1]	*				
STEUERN VARIABLE [1]	*	*	*	*	

[1] Tätigkeiten, die über mehrere Systemkontrollpunkte verteilt werden können

[2] Pro Systemkontrollpunkt maximal ein Baustein

[3] Nach "Komprimieren im STOP" ist NEUSTART erforderlich.

11.4 Serielle Kopplung PG – AG über 1. oder 2. serielle Schnittstelle

Für die serielle Kopplung PG – AG existieren folgende Möglichkeiten:

- Direkte Verbindung CPU – über die Standard-Steckleitung,
- Verbindung zum PG über den Koordinator KOR C . Dabei wird das PG über die Steckleitung mit dem Koordinator verbunden. Damit ist die 1. serielle Schnittstelle nicht mehr benutzbar.
- Verbindung zum PG über einen PG-Multiplexer 757. Die zulässigen Steckverbindungen können Sie dem Systemhandbuch S5-135U/155U /2/ entnehmen.
- Verbindung zum PG über SINEC H1/L2/L1 und "Affenschaukel"; dabei können auch KOR C oder PG-Multiplexer zwischengeschaltet werden.

11.5 Parallelbetrieb von zwei seriellen PG-Schnittstellen

Die zweite Schnittstelle der CPU 928B (SI 2) können Sie ebenso wie die erste Schnittstelle als **PG-Schnittstelle** nutzen.

Um Ihr PG über diese Schnittstelle ankoppeln zu können, müssen Sie zusätzlich zu Ihrer CPU 928B das PG-Schnittstellenmodul bestellen (die Bestell-Nummer dazu finden Sie im Systemhandbuch S5-135U/155U /2/).

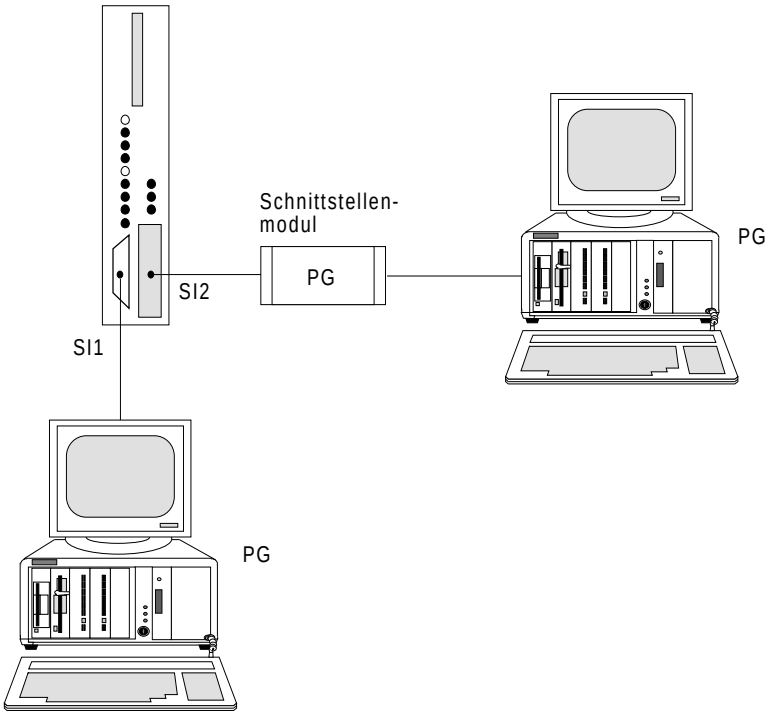


Bild 11-2 Zweite Schnittstelle als PG-Schnittstelle nutzen

Sämtliche PG-Funktionen stehen Ihnen auf beiden Schnittstellen zur Verfügung. In den folgenden Abschnitten finden Sie nur die Informationen, die Sie benötigen, wenn Sie gleichzeitig an beiden Schnittstellen mit Programmiergeräten oder OPs arbeiten.

Konfigurations-Beispiele

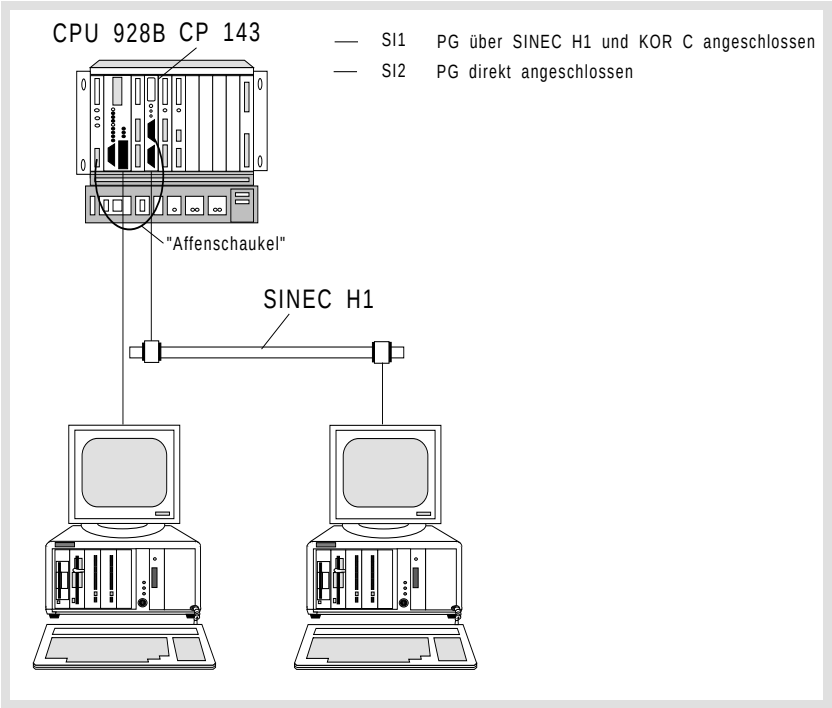


Bild 11-3 1. Konfigurations-Beispiel

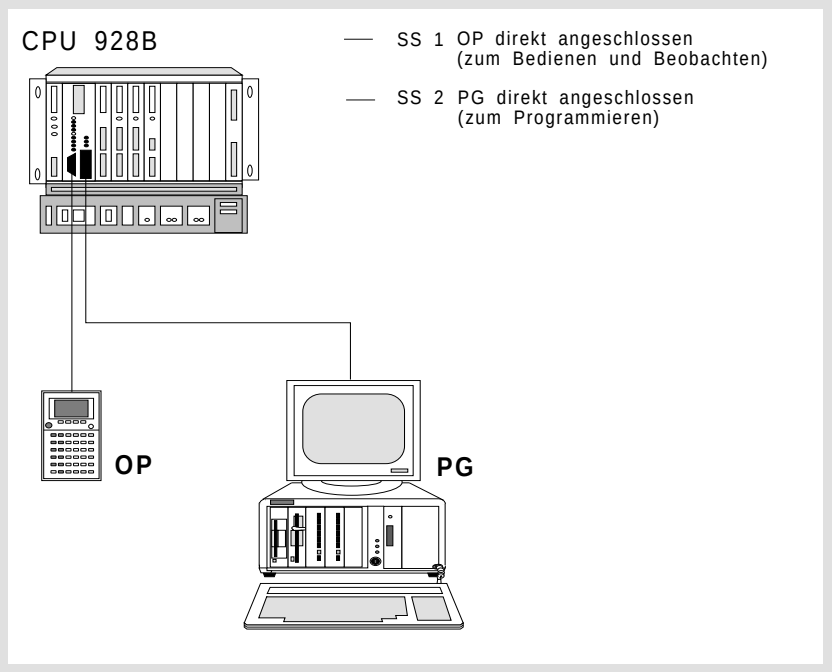


Bild 11-4 2. Konfigurations-Beispiel

11

11.5.1 Inbetriebnahme

Um die zweite Schnittstelle der CPU 928B als PG-Schnittstelle in Betrieb zu nehmen, gehen Sie in folgenden Schritten vor:

Schritt	Aktion
1	Bauen Sie das PG-Modul in die CPU 928B ein. (lesen Sie die Anleitungen dazu bitte im Anhang nach
2	Schließen Sie das PG an die serielle Schnittstelle SI2 an.

11.5.2 Betrieb

Wenn Sie die 2. serielle Schnittstelle als PG-Schnittstelle einsetzen, steht auf jeder Schnittstelle der volle Funktionsumfang der Standard-PG-Schnittstelle zur Verfügung. Dies gilt solange, wie die einzelnen Funktionen sich gegenseitig nicht beeinflussen, d. h. sequentiell nacheinander aufgerufen werden.

Zum Verständnis der Ausnahmen ist es sinnvoll, die PG-Funktionen in drei Gruppen zu unterteilen:

Gruppe	Kennzeichen
kurzlaufende Funktionen	Funktionen, die einen Auftrag ausführen und anschließend beendet sind. (z. B. "Übertragen", "Löschen" usw.)
langlaufende Funktionen	Funktionen, die eine feste Anzahl von Aufträgen bearbeiten: - "Steuern", - "Bearbeitungskontrolle".
zyklische Funktionen	Die Funktionen führen einen Auftrag immer wieder aus, bis sie vom Anwender beendet werden: - "Status Baustein", - "Status Variablen", - "Steuern Variablen".

Vorsicht
Bei langlaufenden und zyklischen Funktionen müssen Sie den Aufruf dieser Funktionen auf beiden PGs koordinieren.

In der nachfolgenden Tabelle sind die Funktionspaare aufgelistet, die Sie **nicht parallel** betreiben können.

Tabelle 11-3 Funktionen, die nicht parallel auf zwei PGs ablaufen

Am ersten PG läuft die Funktion:	**Diese Funktion ist am zweiten PG *nicht möglich*:**
"Steuern"	jede Funktion
"Bearbeitungskontrolle"	jede Funktion
eine "Status"-Funktion"	"Steuern"
eine "Status"-Funktion"	"Bearbeitungskontrolle"
eine "Status"-Funktion"	"Urlöschen"
"Status" auf nicht bearbeiteten oder langlaufenden Baustein	jede Funktion

Wenn Sie dies mißachten, meldet das zweite PG einen Fehler, z. B.: *"AS-Funktion gesperrt: laufende Funktion"*.

Die gleiche Fehlermeldung oder *"Zeitüberlauf bei Datenaustausch mit AG"* erscheint, wenn die CPU 928B gerade Funktionen des anderen PGs bearbeitet, welche den Zugriff Ihres PG innerhalb der Zeitüberwachung verhindern. Ihre Eingabe wird dann abgewiesen. Wiederholen Sie Ihre Eingabe, nachdem die Funktionen des anderen PGs abgeschlossen sind.

Hinweis
Durch unterschiedlichen Leistungs- und Funktionsumfang sind Zeitüberwachung und Fehlermeldeverhalten nicht bei allen PGs und OP gleich.

Wenn Sie gleichzeitig an beiden PGs die Funktion "Speicherausbau" aktivieren, kann es zu falschen Anzeigen kommen.

Vorsicht
Wenn Sie gleichzeitig online an beiden PGs Bausteine eingeben, korrigieren oder löschen, so müssen Sie beachten, daß die Bausteine nicht vor dem Zugriff durch das jeweils andere PG geschützt sind.
"Status" auf nicht angesprungenen Baustein oder "Status" im STOP blockiert die andere Schnittstelle für alle Funktionen.

11.5.3 Ablauf bei bestimmten Betriebsfällen

Paralleler Betrieb bei kurzlaufenden Funktionen

Wenn Sie an beiden Schnittstellen gleichzeitig mit Programmiergeräten arbeiten, versuchen beide PGs, unabhängig voneinander Ihre jeweiligen Funktionen auszuführen. Solange sie ihre Aufträge zeitlich versetzt an die CPU übertragen, werden diese in der Reihenfolge ihres Eintreffens nacheinander bearbeitet.

Es kann aber vorkommen, daß die CPU 928B zwei Aufträge entweder gleichzeitig erhält oder einen Auftrag vom zweiten PG erhält, während noch ein Auftrag des ersten PGs läuft.
Da die gleichzeitige Bearbeitung nicht möglich ist, werden diese beiden Aufträge nacheinander bearbeitet; allerdings sind die Wartezeiten für den zweiten Auftrag meistens so gering, daß sie für den Anwender kaum in Erscheinung treten.

Es ergibt sich somit für gleichzeitige Aufträge folgender Ablauf:

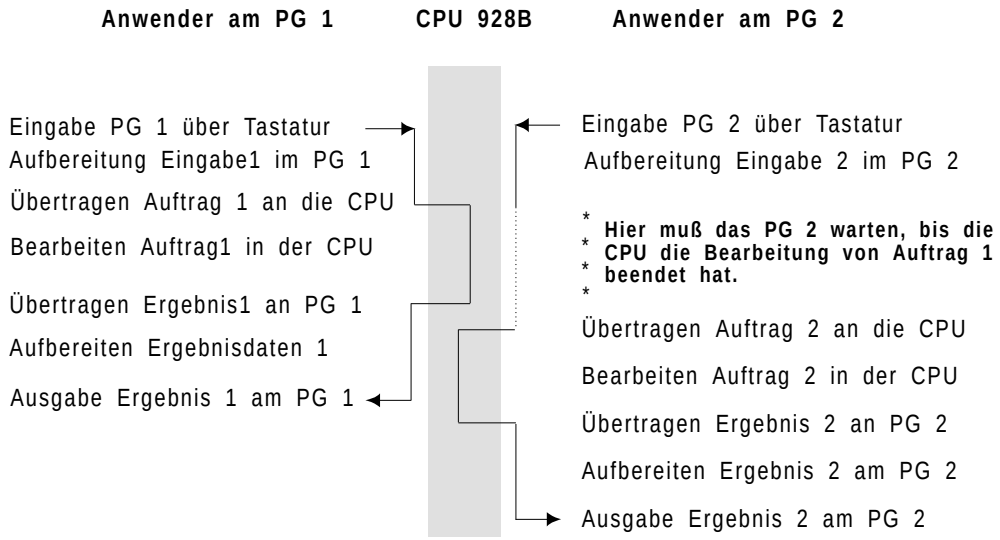


Bild 11-5 Zeitlicher Ablauf bei gleichzeitigen Aufträgen

Aus diesem Ablauf ist zu ersehen, daß Sie zwar an beiden PGs unabhängig voneinander arbeiten können, daß aber eine gegenseitige Beeinflussung auftritt.
So kann es passieren, daß beide PG gleichzeitig denselben Baustein bearbeiten oder über ein PG ein Baustein gelöscht wird, den das andere PG gerade bearbeitet.
Sie müssen also bei jeder Bearbeitung berücksichtigen, inwieweit Sie das Verhalten des anderen PG beeinflussen.

Paralleler Betrieb bei langlaufenden Funktionen

Die langlaufenden Funktionen "Steuern" und "Bearbeitungskontrolle" können keine andere Funktion unterbrechen und können auch von keiner anderen Funktion unterbrochen werden. Sie dürfen damit auch nicht parallel ausgeführt werden, d. h. sie werden wie ein Standardauftrag als ein Block behandelt.

Paralleler Betrieb bei zyklischen Funktionen

Zyklische Funktionen können parallel sowohl zu zyklischen als auch zu kurzlaufenden Funktionen ausgeführt werden. Als Beispiel wird hier der Standardablauf für die Funktion "Status Variablen" dargestellt.

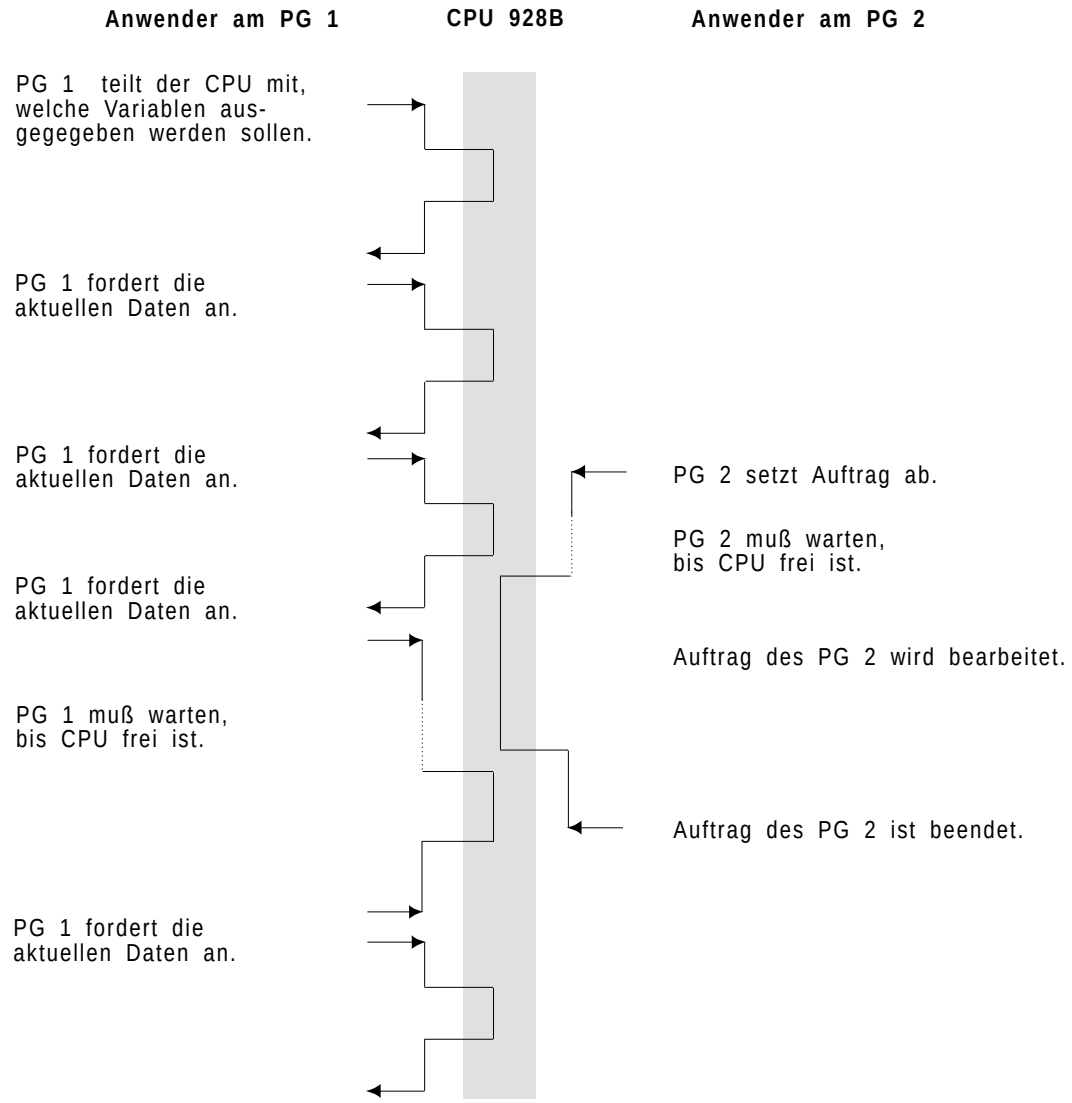


Bild 11-6 Typischer zeitlicher Ablauf einer zyklischen Funktion mit paralleler, kurzlaufender Funktion

Um einem zweiten PG die Möglichkeit zu bieten, auch einen Auftrag an die CPU zu senden, wird die Statusfunktion zwischen zwei Anforderungen unterbrochen und nach dem eingeschobenen Auftrag wieder fortgesetzt. Da die unterbrechende Funktion zu ihrer Bearbeitung CPU-Leistung benötigt, muß die gesamte CPU-Systemleistung auf beide Funktionen aufgeteilt werden, z. B. werden bei der Funktion "Status Variablen" die auszugebenden Daten etwas langsamer aktualisiert.

Bei gleichzeitigem Arbeiten an beiden PGs ergibt sich der in Bild 11.7 dargestellte Ablauf.

Dieser gilt auch für den Fall, daß auf beiden PGs zyklische Funktionen ablaufen; die beiden PGs wechseln sich dann beim Zugriff auf die CPU ab.

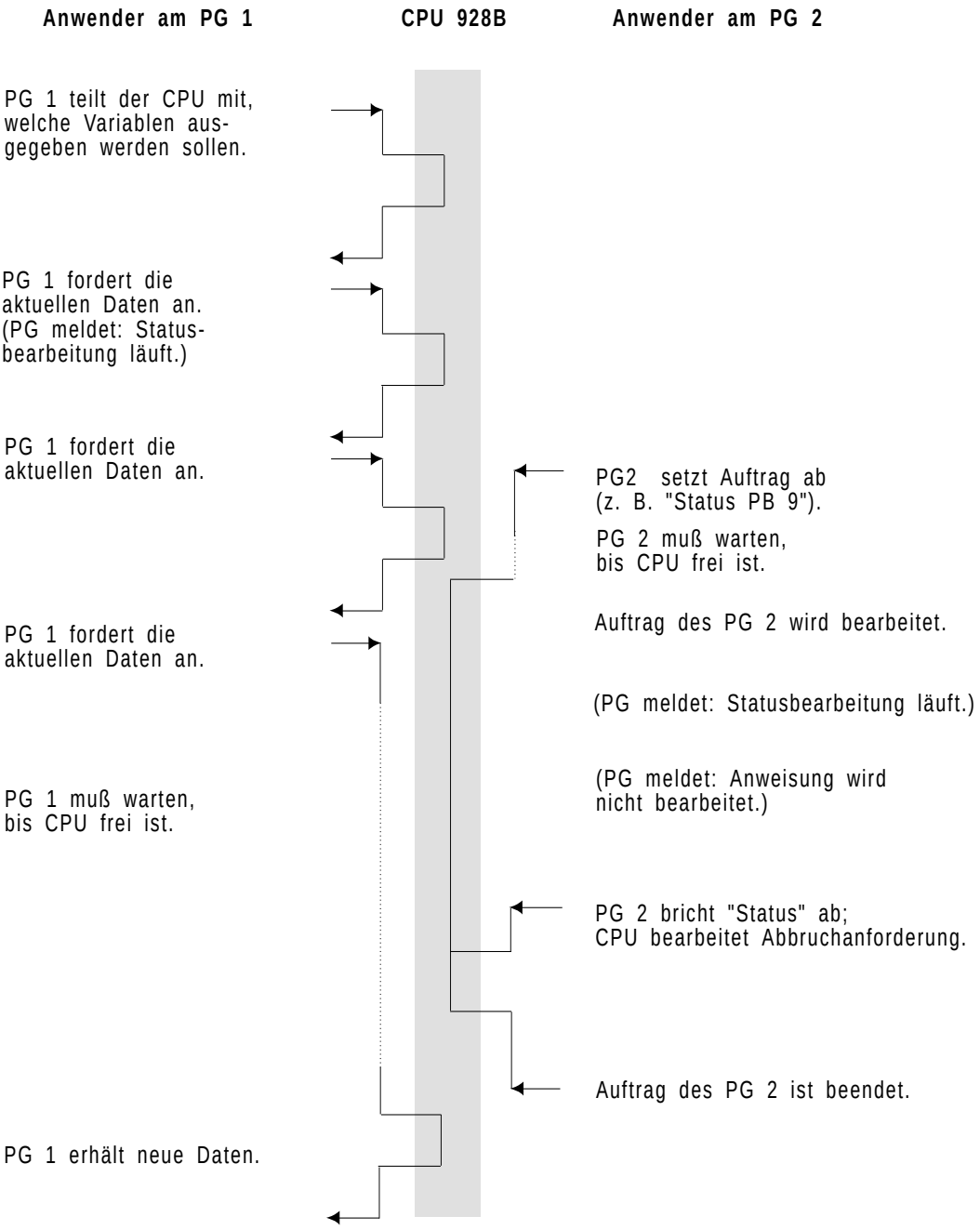


Bild 11-7 Zeitlicher Ablauf zweier paralleler zyklischer Funktionen

Besonderheit bei zyklischen Funktionen auf beiden PG

Blockiert die unterbrechende Funktion die CPU 928B ("Status" auf einen Baustein, der nicht ausgeführt wird), so wird die unterbrochenen Funktion ebenfalls blockiert. Sie wird erst dann wieder fortgesetzt, wenn die unterbrechende Funktion beendet wird.

Bei gleichzeitigem Arbeiten an beiden PGs ergibt sich dann folgender Ablauf:

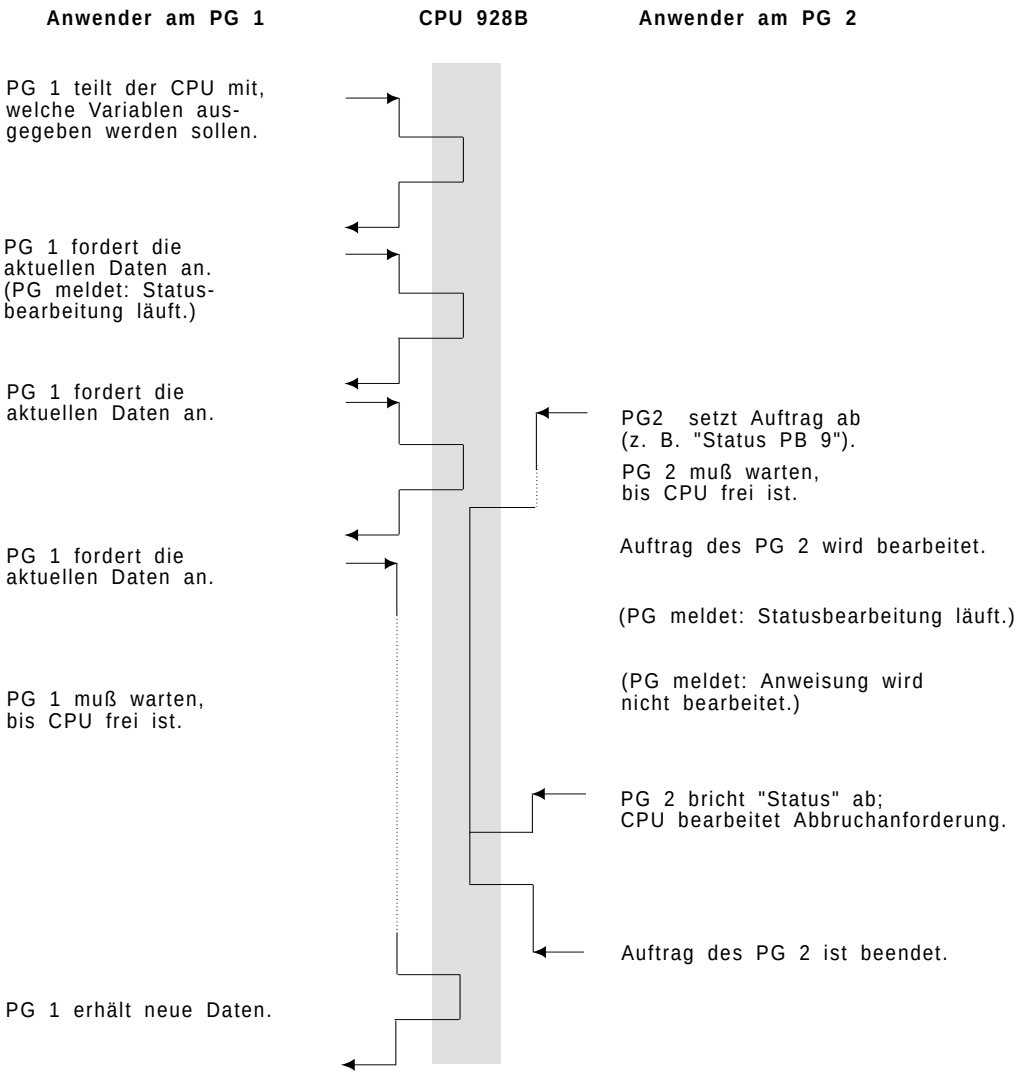


Bild 11-8 Zeitlicher Ablauf, wenn eine Funktion die CPU 928B blockiert

Allgemeine Hinweise

Wird an **einer** Schnittstelle "Status Variablen", "Steuern Variablen" (bei der Statusanzeige) oder "Status" ausgegeben und an der **anderen** Schnittstelle "Speicher komprimieren", "Baustein löschen" oder "Baustein übertragen" durchgeführt, so kann es zu einer falschen Anzeige in der Statusanzeige kommen.

Anhang 12

Inhalt von Kapitel 12

12

Anhang

Dieses Kapitel gibt Ihnen einige zusätzliche Informationen zur CPU 928B wie Laufzeitenvergleich mit CPU 922, CPU 928 und CPU 928B, Fehler- und Ebenenkennungen und andere für die Fehlerdiagnose hilfreiche Daten und Erläuterungen

Anhang 1: Technische Daten der CPUs im AG S5-135U

Operation / Bearbeitung	CPU 922	CPU 928	CPU 928B
typische Befehlsausführungszeiten für Bitbefehle:			
mit M, E, A D Formaloperand	 22 µs 37 µs 46 µs	 1 µs 34 µs 25 µs	 0,57 µs 3,4 µs 2,4 µs
typische Befehlsausführungszeiten für Wortbefehle:			
- Ladeoperationen L MB (Byte) L MW (Wort) L MD (Doppelwort) - Festpunktarithmetik - Gleitpunktarithmetik	 15 µs 15 µs 20 µs 26 ... 50 µs 51 ... 86 µs	 12 µs 12 µs 16 µs 13 ... 24 µs 29 ... 69 µs	 0,81 µs 0,94 µs 1,6 µs 0,94 ... 10 µs 9,1 ... 23 µs
zyklische Programmbearbeitung (Einzelprozessorbetrieb)			
Grundlast bei Aufruf OB 1/FB 0:	107/119 µs	147/149 µs	170/172 µs
Zuschlag für die Prozeßabbild-aktualisierung in Abhängigkeit von der Anzahl der E/A-Bytes (n) mit 0 < n ≤128	33 µs + n * 6 µs	18 µs + n * 1,58 µs	18 µs + n * 2,13 µs
Zuschlag für die Koppelmerker-übertragung in Abhängigkeit von der Anzahl der Koppelmerker (n) mit 0 < n ≤ 256	35 µs + n * 6,5 µs	19 µs + n * 1,84 µs	n ≤ 128: 18 µs + n * 2,38 µs n > 128: 36 µs + n * 2,38 µs
Zuschlag für Zeitzellenbearbeitung in Abhängigkeit von der Zeitenblocklänge (ZBL) ZBL =0 ZBL #0 n = Anzahl der laufenden Zeitzellen (Raster: 10 ms)	alle 2,5 ms 50 µs 60 µs + ZBL * 1,56 µs + n * 1,24 µs	alle 10 ms 5 µs 200 µs + n * 0,35 µs (bei 0 <n ≤ 128) 400 µs + n * 0,35 µs (bei 128 <n≤256)	alle 10 ms 1 µs 20 µs + ZBL * 1 µs (keine Unterschiede zwischen laufenden und nicht laufenden Zeitzellen)

Operation / Bearbeitung	**CPU 922**	**CPU 928**	**CPU 928B**
alarmgesteuerte Programmbearbeitung			
Verlängerung der Zykluszeit durch Einschachtelung eines leerenOB 2 (ohne STEP-5-Operationen) an einer Bausteingrenze	367 µs	330 µs	492 µs
Reaktionszeit	300 µs	280 µs	297 µs
zeitgesteuerte Programmbearbeitung			
Verlängerung der Zykluszeit durch Einschachtelung eines leeren OB 13 (ohne STEP-5-Operationen) an einer Befehlsgrenze	375 µs	340 µs für den ersten Weckalarm-OB 180 µs für jeden weiteren, zum gleichen Zeitpunkt fälligen Weckalarm-OB	440 µs für den ersten Weckalarm-OB 200 µs für jeden weiteren, zum gleichen Zeitpunkt fälligen Weckalarm-OB
Zeittakt für den Aufruf des zeitgesteuerten Programms (Weckalarme OB 10 bis OB 18)	100 ms	10, 20, 50, 100, 200, 500 ms, 1, 2, 5 sec	10, 20, 50, 100, 200, 500 ms, 1, 2, 5 sec
Auflösungszeiten für uhrzeitgesteuerten Weckalarm (OB 9)	–	–	minütlich, stündlich, täglich, wöchentlich, monatlich, jährlich, einmalig
Auflösungszeit für Verzögerungsalarm (OB 6)	–	–	1 ms
Zykluszeitüberwachung			
Voreinstellung einstellbar zwischen triggerbar	150 ms 1 ... 4000 ms ja	150 ms 1 ... 6000 ms ja	150 ms 1 ... 13000 ms ja

Operation / Bearbeitung	CPU 922	CPU 928	CPU 928B
Speichergrößen			
Größe des Anwenderspeichermoduls (in K byte)	64	64	64
Größe des Speichers für Datenbausteine (DB-RAM, in K byte)	ca. 22,2	ca. 46,6	ca. 46,6
Zeit- und Zählzellen, Merker			
Anzahl der Zeit- und Zählzellen	je 128	je 256	je 256
Anzahl der Merker	2048 Merker	2048 Merker	2048 Merker + 8192 S-Merker

Begriffserläuterungen

Grundlast

Als Grundlast wird derjenige Teil der zyklischen Systemlaufzeit bezeichnet, der ohne Prozeßabbildaktualisierung, Koppelmerkerübertragung und ohne Unterbrechungen durch Alarme oder Fehler durchlaufen wird.

Reaktionszeit

Als Reaktionszeit wird die Zeit vom Aktivieren der Programmbearbeitungsebene PROZESSALARM bis zur Bearbeitung der ersten Operation im OB 2 bezeichnet. Es wird dabei vorausgesetzt, daß der OB 2 **sofort** nach Erkennen des Prozeßalarms aufgerufen werden kann. Muß hingegen auf die nächste Befehls- oder Bausteingrenze **gewartet** werden, verlängert sich die Reaktionszeit entsprechend.

Anhang 2: Fehlerkennungen

Fehlerkennungen im Systemdatum BS 3 und BS 4

BS 3	BS 4	Erläuterung
		Aufbau der Bausteinadreßlisten (Auswertung des DB 0)
8001H	yyyyH	Falsche Bausteinlänge yyyy = Adresse des Bausteins mit falscher Länge
8002H	yyyyH	Berechnete Endadresse des Bausteins im Speicher falsch yyyy = Bausteinadresse
8003H	yyyyH	Ungültige Bausteinkennung yyyy = Adresse des Bausteins mit falscher Kennung
8004H	yyyyH	Zu große Organisationsbausteinnummer (erlaubt: OB 1 bis OB 39) yyyy = Adresse des Bausteins mit falscher Nummer
8005H	yyyyH	Datenbausteinnummer 0 (erlaubt: DB 1 bis DB 255) yyyy = Adresse des Bausteins mit falscher Nummer
		Aufbau der Adreßlisten für die Prozeßabbild-Aktualisierung (Auswertung des DB 1)
0410H	yyyyH	Unzulässige Kennung: - Kopfkennung fehlt oder fehlerhaft (korrekt KC MASK01) - Kennung unzulässig (zulässig KH DE00, DA00, CE00, CA00, BB00) - Endekennung fehlt oder fehlerhaft (korrekt KH EEEE) yyyy = unzulässige Kennung
0411H	yyyyH	"Digitale Eingänge" , Anzahl Adressen unzulässig (zulässig 0 ... 128) yyyy = unzulässige Anzahl Adressen
0412H	yyyyH	"Digitale Ausgänge" , Anzahl Adressen unzulässig (zulässig 0 ... 128) yyyy = unzulässige Anzahl Adressen
0413H	yyyyH	"Koppelmerker-Eingänge" , Anzahl Adressen unzulässig (zulässig 0 ... 256) yyyy = unzulässige Anzahl Adressen
0414H	yyyyH	"Koppelmerker-Ausgänge", Anzahl Adressen unzulässig (zulässig 0 ... 256) yyyy = unzulässige Anzahl Adressen
0415H	yyyyH	Ungültige Anzahl Zeitzellen (erlaubt: 256) yyyy = unzulässige Anzahl Zeitzellen
0419H	yyyyH	Quittungsverzug bei digitalen Eingängen yyyy = Adresse des nicht quittierten Eingangsbytes
041AH	yyyyH	Quittungsverzug bei digitalen Ausgängen yyyy = Adresse des nicht quittierten Ausgangsbytes
041BH	yyyyH	Quittungsverzug bei Koppelmerker-Eingang yyyy = Adresse des nicht quittierten Koppelmerkerbytes
041CH	yyyyH	Quittungsverzug bei Koppelmerker-Ausgang yyyy = Adresse des nicht quittierten Koppelmerkerbytes

BS 3	BS 4	Erläuterung
		Auswertung des DB 2
0421H	DByyH	Datenbaustein nicht geladen yy = Nummer des nicht geladenen Datenbausteins
0422H	FByyH	Funktionsbaustein nicht geladen yy = Nummer des nicht geladenen Funktionsbausteins
0423H	FByyH	Funktionsbaustein nicht erkannt yy = Nummer des nicht erkannten Funktionsbausteins
0424H	FByyH	Funktionsbaustein mit falscher PG-Software geladen yy = Nummer des Funktionsbausteins
0425H	DByyH	Falsche Regler-Datenbaustein-Länge yy = Nummer des Datenbausteins
0426H	-	Für das Verschieben der Regler-DB vom Anwender-EPROM in das DB-RAM ist der Speicherplatz im DB-RAM nicht ausreichend
		Auswertung des DX 0
0431H	yyyyH	Unzulässige Kennung -Kopfkennung fehlt oder fehlerhaft (korrekt KC MASKX0) -Blockkennung unzulässig -Endekennung fehlt oder fehlerhaft (korrekt KH EEEE) yyyy = unzulässige Kennung
0432H	yyyyH	Unzulässiger Parameter yyyy = unzulässiger Parameter
0434H	yyyyH	Nicht erlaubte Anzahl Zeitzellen (erlaubt: 0...256) yyyy = falsche Anzahl Zeitzellen
0435H	yyyyH	Unerlaubte Zyklusüberwachungszeit (erlaubt: 1ms bis 13000ms) yyyy = falsche Zeitgröße
		Auswertung des DX 2
0451H	-	DX-2-Länge (ohne Bausteinkopf)< 4 Wörter ist unzulässig
0452H	yyyyH	DX-2-Länge (ohne Bausteinkopf) ist für Kopplungstyp zu kurz yyyy = Länge DX 2
0453H	yyyyH	Kopplungstyp unzulässig yyyy = Kopplungstyp
0454H	xx00H	Datenkennung für stat. Parametersatz ungültig (ungleich 44H, 58H) xx = Datenkennung
0455H	xxyyH	Baustein für statischen Parametersatz unzulässig xx = Kennung / yy = DB-Nummer
0456H	xxyyH	Statischer Parametersatz nicht vorhanden xx = Kennung / yy = DB-Nummer
0457H	yyyyH	Statischer Parametersatz zu kurz yyyy = Nummer des nicht vorhandenen DW
0458H	xx00H	Datenkennung für dyn. Parametersatz ungültig (ungleich 44H, 58H, 00H) xx = Datenkennung
0459H	xxyyH	Baustein für dyn. Parametersatz unzulässig xx = Kennung / yy = DB-Nummer
0045AH	xx00H	Datenkennung für Sendefach/Auftragsfach ungültig (ungleich 44H, 58H,00H) xx = Datenkennung

BS 3	BS 4	Erläuterung
Auswertung des DX 2 (Fortsetzung)		
045BH	xxyyH	Baustein für Sendefach/Auftragsfach unzulässig xx = Kennung / yy = DB-Nummer
045CH	xx00H	Datenkennung für Empfangsfach ungültig (ungleich 44H, 58H, 00H) xx = Datenkennung
045DH	xxyyH	Baustein für Empfangsfach unzulässig xx = Kennung / yy = DB-Nummer
045EH	xx00H	Datenkennung für Koordinierungsbytes ungültig (ungleich 44H, 58H, 4DH) xx = Kennung
045FH	xxyyH	Baustein für Koordinierungsbytes unzulässig xx = Kennung / yy = DB-Nummer
0460H	xxyyH	Baustein für Koordinierungsbytes nicht vorhanden xx = Kennung / yy = DB-Nummer
0461H	yyyyH	Datenwort für Koordinierungsbytes nicht vorhanden yyyy = Nr. des nicht vorhandenen DW

Fehlerkennungen in AKKU 1 und AKKU 2

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
		REG-FE (Reglerfehler)	
0801H	DByyH	Abtastzeitfehler yy = Nummer des betreffenden Regler-Datenbausteins	OB 34
0802H	DByyH	Regler-Datenbaustein nicht geladen yy = Nummer des nicht geladenen Datenbausteins	
0803H	FByyH	Regler-Funktionsbaustein nicht geladen yy = Nummer des nicht geladenen Funktionsbausteins	
0804H	FByyH	Regler-Funktionsbaustein nicht erkannt yy = Nummer des nicht erkannten Funktionsbausteins	
0805H	FByyH	Regler-Funktionsbaustein mit falscher PG-Software geladen yy = Funktionsbaustein-Nr.	
0806H	DByyH	Falsche Regler-Datenbaustein-Länge yy = Datenbaustein-Nr.	
0880H	00yyH	Quittungsverzug (QVZ) während der Reglerbearbeitung yy = Nr. des E/A-Bytes, das QVZ verursacht hat	
		WECK-FE (Weckfehler)	
1001H	0016H	Weckfehler bei OB 10 (10 ms)	OB 33
	0014H	Weckfehler bei OB 11 (20 ms)	
	0012H	Weckfehler bei OB 12 (50 ms)	
	0010H	Weckfehler bei OB 13 (100 ms)	
	000EH	Weckfehler bei OB 14 (200 ms)	
	000CH	Weckfehler bei OB 15 (500 ms)	
	000AH	Weckfehler bei OB 16 (1 sec)	
	0008H	Weckfehler bei OB 17 (2 sec)	
	0006H	Weckfehler bei OB 18 (5 sec)	
		BCF (Befehlscodefehler)/Substitutionsfehler	
1801H	–	Substitutionsfehler beim Befehl BBS	OB 27
1802H	–	Substitutionsfehler bei BDW, BMW	
1803H	–	Substitutionsfehler bei den Befehlen B= , BI=	
1804H	–	Substitutionsfehler bei den Befehlen L= , = T	
1805H	–	Substitutionsfehler bei den Befehlen U=, UN=, O=, ON=, S= und RB=	
1806H	–	Substitutionsfehler bei den Befehlen RD=, LC=, FR=, SAR=, SE=, SI=, SSV= und SVZ=	

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
		BCF (Befehlscodefehler)/Operationscodefehler	+
1811H	–	Befehl mit unzulässigem Opcode	OB 29
1812H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 68H enthält	
1813H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 78H enthält	
1814H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 70H enthält	
1815H	–	Unzulässiger Opcode bei einem Befehl, bei dem das High-Byte des 1. Befehlswortes den Wert 60H enthält	
		BCF (Befehlscodefehler)/Parameterfehler	
		Unzulässiger Parameter bei:	OB 30
1821H	–	A DB 0, 1, 2	
182BH	–	SPA(B) OB 0	
182CH	–	SPA(B) OB >39: Sonderfunktion nicht vorhanden	
182DH	–	AX DX 0, AX DX 1 und AX DX 2	
182EH	–	L MW / T MW / L PW / T PW / L QW / T QW / L DD / T DD / B MW : 255	
182FH	–	L EW/T EW/L AW/T AW 127	
1830H	–	L MD / T MD 253, 254, 255	
1831H	–	L ED/T ED/L AD/T AD 125, 126, 127	
1832H	–	RLD/RRD/SVD/SLD 33-255	
1833H	–	SLW/SRW/LIR/TIR 16-255	
1834H	–	SES/SEF 32-255	
1835H	–	U=/UN=/O=/ON=/S=/RB=/==/RD=/FR=/SI=/SE=/ SVZ=/SSV=/SAR=/L=/LC=/LW=/T= 0, 127-255	
1836H	–	B=/LD= 0, 126-255	
1837H	–	U S/O S/S S/= S/UN S/ON S/R S Bytenummer > 1023	
1838H	–	U S/O S/S S/= S/UN S/ON S/R S Bitnummer > 7	
1839H	–	L SY/T SY Parameter > 1023	
183AH	–	L SW/T SW Parameter > 1022	
183BH	–	L SD/T SD Parameter >1020	
183CH	–	E DB/EX DX Parameter 0, 1 oder 2 (DB bzw. DX 0, 1, 2 nicht erzeugbar)	
		LZF (Laufzeitfehler)/nicht geladener Baustein	
1A01H	–	Nicht geladener Datenbaustein bei A DB	OB 19
1A02H	–	Nicht geladener Datenbaustein bei AX DX	
1A03H	–	Nicht geladener Baustein bei SPA(B) FB, OB 1 bis OB 39, PB, SB	
1A04H	–	Nicht geladener Baustein bei BA(B) FX	
1A05H	–	Nicht geladener Datenbaustein bei OB 254 bzw. 255	
1A06H	–	Nicht geladener Datenbaustein bei OB 182	
1A07H	–	Nicht geladener Datenbaustein bei OB 150/OB 151	

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
		LZF (Laufzeitfehler)/Lade-bzw.Transferfehler	
1A11H	–	Zugriff mit U/UN D, O/ON D, S/R D, = D auf ein nicht definiertes Datenwort	OB 32
1A12H	–	Transferfehler bei TDR auf ein nicht definiertes Datenwort	
1A13H	–	Transferfehler bei TDL auf ein nicht definiertes Datenwort	
1A14H	–	Transferfehler bei TDW auf ein nicht definiertes Datenwort	
1A15H	–	Transferfehler bei TDD auf ein nicht definiertes Datenwort	
1A16H	–	Ladefehler bei LDR auf nicht definiertes Datenwort	
1A17H	–	Ladefehler bei LDL auf nicht definiertes Datenwort	
1A18H	–	Ladefehler bei LDW auf nicht definiertes Datenwort	
1A19H	–	Ladefehler bei LDD auf nicht definiertes Datenwort	
		LZF (Laufzeitfehler)/sonstige Laufzeitfehler	
		Fehleranzeige von .../durch ... :	OB 31
1A21H	–	E DB, EX DX: Datenbaustein existiert bereits	
1A22H	–	E DB, EX DX: unzulässige Anzahl Datenwörter (< 1 oder > 4091)	
1A23H	–	E DB, EX DX: Speicherplatz im RAM reicht nicht aus	
1A25H	–	BI: unzulässiger Parameter im AKKU 1 (< 1 oder > 125)	
1A29H	–	Klammerstackunter- oder -überlauf nach 'U(', 'O(, ')'	
1A2AH	–	A DB, AX DX: Bausteinlänge im Datenbausteinkopf ist zu klein (Länge <5 Wörter)	
1A2BH	–	Funktionsbaustein ist mit falscher PG-Software geladen	
1A2CH	–	ACR: Kachelnummer in AKKU-1-L ist unzulässig (> 255)	
1A31H	–	OB 254 bzw. OB 255 (Verschieben) oder OB 250: Ziel-Datenbaustein ist bereits im DB-RAM vorhanden	
1A32H	–	OB 254 bzw. OB 255 (Duplizieren): Ziel-Datenbaustein ist bereits im DB-RAM vorhanden	
1A33H	–	OB 254 bzw. OB 255 oder OB250: Speicherplatz im DB-RAM reicht nicht aus	
1A34H	0001H	OB 182: Beschreibung des Datenfeldes ist unzulässig	
1A34H	0100H	OB 182: Adreßbereichs-Typ ist unzulässig	
1A34H	0101H	OB 182: Datenbaustein-Nr. ist unzulässig	
1A34H	0102H	OB 182: "Nummer des ersten Parameterwortes" ist unzulässig	
1A34H	0200H	OB 182: "Quelldatenbaustein-Typ" ist unzulässig	
1A34H	0201H	OB 182: "Quelldatenbaustein-Nummer" ist unzulässig	
1A34H	0202H	OB 182: "Nummer des ersten zu übertragenden Datenwortes in der Quelle" ist unzulässig	
1A34H	0203H	OB 182: als Länge des Quelldatenbaustein ist im Bausteinkopf ein Wert < 5 Wörter eingetragen	
1A34H	0210H	OB 182: "Zieldatenbaustein-Typ" ist unzulässig	
1A34H	0211H	OB 182: "Zieldatenbaustein-Nummer" ist unzulässig	
1A34H	0212H	OB 182: "Nummer des ersten zu übertragenden Datenwortes im Ziel" ist unzulässig	
1A34H	0213H	OB 182: als Länge des Zieldatenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen	

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
LZF (Laufzeitfehler)/sonstige Laufzeitfehler (Fortsetzung)			
		Fehleranzeige von .../durch ... :	
1A34H	0220H	OB 182: "Anzahl zu übertragender Datenworte" ist unzulässig (= 0 oder > 4091)	OB 31
1A34H	0221H	OB 182: Quelldatenbaustein ist zu kurz	
1A34H	0222H	OB 182: Zieldatenbaustein zu kurz	
1A34H	0223H	OB 182: Zieldatenbaustein ist im EPROM gespeichert	
1A35H	–	OB 250: Nummer des Übergabebausteins ist unzulässig	
1A36H	–	OB 250: unterschiedliche Länge bei DB x und DB x+1 bzw. DX x und DX x+1	
1A3AH	–	OB 221: unzulässiger Wert für die neue Zykluszeit (Zykluszeit <1 ms oder > 13 000 ms)	
1A3BH	–	OB 223: Anlaufarten der am Mehrprozessorbetrieb beteiligten CPUs sind unterschiedlich	
1A41H	–	OB 240, OB 241 oder OB 242: Schieberegister- oder Datenbaustein-Nummer ist unzulässig (Nr. < 192 oder > 255)	
1A42H	–	OB 241: Schieberegister ist nicht initialisiert	
1A43H	–	OB 240: Speicherplatz im DB-RAM reicht aus	
1A44H	–	OB 240: Datenwort DW 0 des Datenbausteins hat nicht den Inhalt '0'	
1A45H	–	OB 240: Schieberegisterlänge in DW 1 ist unzulässig (nicht zwischen 2 und 256)	
1A46H	–	OB 240: Zeigerposition ist unzulässig oder Zeigeranzahl ist > 5	
1A47H	–	OB 120: Wert in AKKU 1 oder AKKU-2-L ist unzulässig	
1A48H	–	OB 122: Wert in AKKU 1 ist unzulässig	
1A49H	–	OB 110: Wert in AKKU 1 oder AKKU-2-L ist unzulässig	
1A4AH	–	OB 121: Wert in AKKU 1 oder AKKU-2-L ist unzulässig	
1A4BH	–	OB 123: Wert in AKKU 1 ist unzulässig	
1A4CH	0001H	OB 150: Funktionsnummer ist unzulässig (= 0 oder > 2)	
1A4CH	0100H	OB 150: Adreßbereichs-Typ ist unzulässig	
1A4CH	0101H	OB 150: Datenbaustein-Nr. ist unzulässig	
1A4CH	0102H	OB 150: "Nummer des ersten Datenfeldwortes" ist unzulässig	
1A4CH	0103H	OB 150: als Länge des Datenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen	
1A4CH	0201H	OB 150: Jahresangabe im Datenfeld ist unzulässig	
1A4CH	0202H	OB 150: Monatsangabe im Datenfeld ist unzulässig	
1A4CH	0203H	OB 150: Monatstagangabe im Datenfeld ist unzulässig	
1A4CH	0204H	OB 150: Wochentagangabe im Datenfeld ist unzulässig	
1A4CH	0205H	OB 150: Stundenangabe im Datenfeld ist unzulässig	
1A4CH	0206H	OB 150: Minutenangabe im Datenfeld ist unzulässig	
1A4CH	0207H	OB 150: Sekundenangabe im Datenfeld ist unzulässig	
1A4CH	0208H	OB 150: Wert "1/100 Sekunde" im Datenfeld ist ungleich 0	
1A4CH	0209H	OB 150: Datenfeldwort 3 /Bit 0 bis 3 ist ungleich 0	
1A4CH	020AH	OB 150: Stunden-Format ist ungleich der Einstellung bei OB 151	
1A4DH	0001H	OB 151: Funktions-Nummer ist unzulässig (= 0 oder > 2)	
1A4DH	0100H	OB 151: Adreßbereichs-Typ ist unzulässig	
1A4DH	0101H	OB 151: Datenbaustein-Nr. ist unzulässig	

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
		LZF (Laufzeitfehler)/sonstige Laufzeitfehler (Fortsetzung)	
		Fehleranzeige von .../durch ... :	
1A4DH	0102H	OB 151: "Nummer des ersten Datenfeldwortes" ist unzulässig	OB 31
1A4DH	0103H	OB 151: als Länge des Datenbausteins ist im Bausteinkopf ein Wert < 5 Wörter eingetragen	
1A4DH	0201H	OB 151: Jahresangabe im Datenfeld ist unzulässig	
1A4DH	0202H	OB 151: Monatsangabe im Datenfeld ist unzulässig	
1A4DH	0203H	OB 151: Monatstagangabe im Datenfeld ist unzulässig	
1A4DH	0204H	OB 151: Wochentagangabe im Datenfeld ist unzulässig	
1A4DH	0205H	OB 151: Stundenangabe im Datenfeld ist unzulässig	
1A4DH	0206H	OB 151: Minutenangabe im Datenfeld ist unzulässig	
1A4DH	0207H	OB 151: Sekundenangabe im Datenfeld ist unzulässig	
1A4DH	0208H	OB 151: Wert "1/100 Sekunde" im Datenfeld ungleich 0	
1A4DH	0209H	OB 151: Auftragsart im Datenfeld ist unzulässig (> 7)	
1A4DH	020AH	OB 151: Stunden-Format ist ungleich der Einstellung bei OB 150	
1A4EH	0001H	OB 152: Funktions-Nr. ist unzulässig (ungleich 0 bis 3 oder ungleich 8 oder ungleich 15)	
1A4FH	0001H	OB 153: Funktions-Nr. ist unzulässig (=0 oder <0)	
1A4FH	0002H	OB 153: Verzögerungszeit ist unzulässig	
1A50H	–	LRW, TRW: Die errechnete Speicheradresse < BR + Konstante> liegt nicht im Bereich "0 .. EDFFH" (s. Kap. 9)	
1A51H	–	LRD, TRD: Die errechnete Speicheradresse < BR + Konstante> liegt nicht im Bereich "0 .. EDFEH" (s. Kap. 9)	
1A52H	–	TSG, LB GB, LW GW, TB GB, TW GW: Die errechnete Linearadresse < BR + Konstante> liegt nicht im Bereich "0 .. EFFFH"	
1A53H	–	LB GW, LW GD, TB GW, TW GD: Die errechnete Linearadresse < BR + Konstante> liegt nicht im Bereich "0 .. EFFEH"	
1A54H	–	LB GD, TB GD: Die errechnete Linearadresse < BR + Konstante> liegt nicht im Bereich "0 .. EFFCH"	
1A55H	–	TSC, LB CB, LW CD, TB CW, TW CD: Die errechnete Kacheladresse < BR + Konstante> liegt nicht im Bereich "F400H .. EBFFH"	
1A56H	–	LB CW, LW CD, TB CW, TW CD: Die errechnete Kacheladresse < BR + Konstante> liegt nicht im Bereich "F400H .. FFFEH"	
1A57H	–	LB CD, TB CD: Die errechnete Kacheladresse < BR + Konstante> liegt nicht im Bereich "F400H .. FBFCH"	

AK-KU-1-L	AK-KU-2-L	Erläuterung	aufgerufener OB
LZF (Laufzeitfehler)/sonstige Laufzeitfehler (Fortsetzung)			
1A58H 1A59H	– –	**Fehleranzeige von .../durch ... :** TNW/TNB: Der Quellblock liegt nicht vollständig in einem dieser Bereiche: 0000 .. 7FFF Anwenderspeicher (siehe Kapitel 9) 8000 .. DD7F Datenbaustein-RAM DD80.. E3FF DB 0 E400 .. E7FF S-Merker E800 .. EDFF System-Daten (BA, BB, BS, BT, Z, T) EE00 .. EFFF Merker, Prozeßabbild F000 .. FFFF Peripherie TNW/TNB: Der Zielblock liegt nicht vollständig in einem dieser Bereiche: 0000 .. 7FFF Anwenderspeicher (siehe Kapitel 9) 8000 .. DD7F Datenbaustein-RAM DD80.. E3FF DB 0 E400 .. E7FF S-Merker E800 .. EDFF System-Daten (BA, BB, BS, BT, Z, T) EE00 .. EFFF Merker, Prozeßabbild F000 .. FFFF Peripherie	OB 31
QVZ (Quittungsverzug)			
1E23H	yyyyH	Quittungsverzug (QVZ) im Anwenderprogramm bei Zugriff auf Peripherie yyyy = QVZ-Adresse	OB 23
1E25H 1E26H 1E27H 1E28H	yyyyH yyyyH yyyyH yyyyH	Quittungsverzug bei der Ausgabe des Prozeßabbildes der digitalen Ausgänge yyyy = Adresse des nicht quittierten Ausgangsbytes Quittungsverzug beim Aktualisieren des Prozeßabbildes der digitalen Eingänge yyyy = Adresse des nicht quittierten Eingangsbytes Quittungsverzug beim Aktualisieren der Koppelmerker-Ausgänge yyyy = Adresse des nicht quittierten Koppelmerkerbytes Quittungsverzug beim Aktualisieren der Koppelmerker-Eingänge yyyy = Adresse des nicht quittierten Koppelmerkerbytes	OB 24
ADF (Adressierfehler)			
1E40H	yyyyH	Adressierfehler (ADF) im Anwenderprogramm yyyy = ADF-Adresse	OB 25

Anhang 3: STEP-5-Operationen, die in der CPU 928B nicht vorhanden sind

Beachten Sie bitte, daß die folgenden STEP-5-Operationen der CPU 946/947 und CPU 948 in der CPU 928B **nicht ablauffähig** sind:

Befehl		Funktion
BAS		Befehlsausgabe sperren
BAF		Befehlsausgabe freigeben
P	E, A, M, Z, T, D, BA, BB, BS, BT	Prüfe Bit auf ’1’
PN	E, A, M, Z, T, D, BA, BB, BS, BT	Prüfe Bit auf ’0’
SU	E, A, M, Z, T, D, BA, BB, BS, BT	Setze Bit unbedingt
RU	E, A, M, Z, T, D, BA, BB, BS, BT	Rücksetze Bit unbedingt
LIM		Lade Interrupt-Maske
SIM		Setze Interrupt-Maske
UBE		Unterbrechungsbaustein-Ende
STW		Stoppbefehl der Weckalarm-bearbeitung
AFS		Adressierfehler-Interrupt sperren
AFF		Adressierfehler-Interrupt freigeben
AAF		Anforderungsalarmbearbeitung freigeben
AAS		Anforderungsalarmbearbeitung sperren

Anhang 4: Kennungen der Programmbearbeitungsebenen

Die Kennungen entsprechen den im USTACK unter **EBENE** eingetragenen Kennungen (hexadezimal).

Kennung	Ebene
0002H	Neustart
0004H	Zyklus
0006H	Weckalarm 5 sec
0008H	Weckalarm 2 sec
000AH	Weckalarm 1 sec
000CH	Weckalarm 500 ms
000EH	Weckalarm 200 ms
0010H	Weckalarm 100 ms
0012H	Weckalarm 50 ms
0014H	Weckalarm 20 ms
0016H	Weckalarm 10 ms
0018H	Zeitauftrag
001AH	nicht belegt
001CH	Regelung
001EH	nicht belegt
0020H	Verzögerungsalarm
0022H	nicht belegt
0024H	Prozeßalarm
0026H	nicht belegt
0028H	Manueller Neustart mit Gedächtnis
002AH	Automatischer Neustart mit Gedächtnis
002CH	Abbruch
002EH	Schnittstellenfehler
0030H	Weckfehler
0032H	Reglerfehler
0034H	Zyklusfehler
0036H	nicht belegt
0038H	Befehlscodefehler
003AH	Laufzeitfehler
003CH	Adressierfehler
003EH	Quittungsverzug
0040H	nicht belegt
0042H	nicht belegt
0044H	Manueller Wiederanlauf
0046H	Automatischer Wiederanlauf

Anhang 5: Beispiel "USTACK-Auswertung"

Dieses (stark vereinfachte) Beispiel verdeutlicht Ihnen eine mögliche Vorgehensweise bei der Auswertung des USTACK.

Beachten Sie dazu auch das Abschnitt 5.4 "Steuerbits und Unterbrechungsstack"!

Ausgangspunkt

Die CPU hat die zyklische Programmbearbeitung abgebrochen und ist in den Stoppzustand übergegangen.

Fehleranalyse

Um die Ursache herauszufinden, wählen Sie am Programmiergerät die Online-Funktion "Ausgabe USTACK" an.

Als erstes werden am PG die Steuerbits ausgegeben:

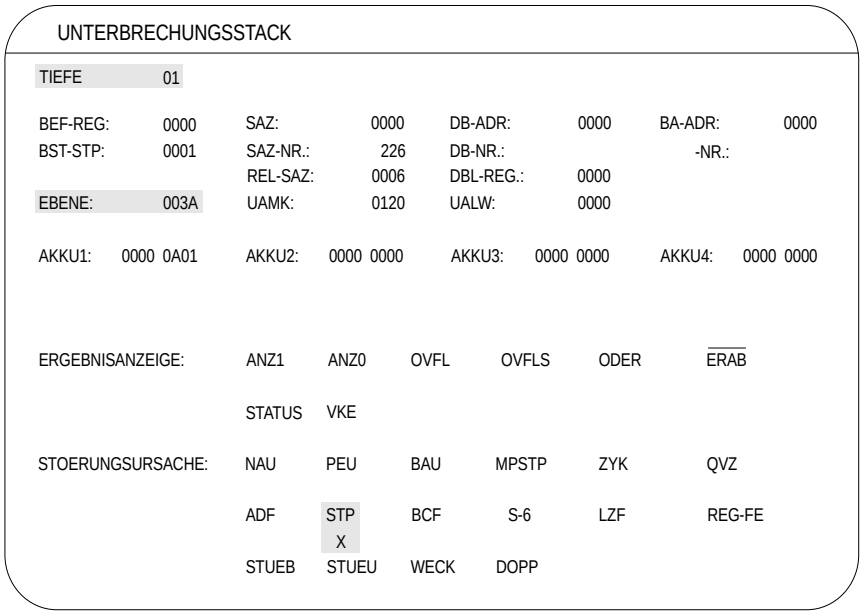
S T E U E R B I T S

>>STP<< X	STP-6	FE-STP	BARBEND	PG-STP	STP-SCH	STP-BEF X	MP-STP
>>ANL<<	ANL-6	NEUSTA X	M W A	A W A	ANL-2	NEUZU X	MWA-ZUL X
>>RUN<<	RUN-6	EINPROZ X	BARB	OB1GEL X	FB0GEL	OBPROZA	OBWECKA
32KWRAM	16KWRAM X	8KWRAM	EPROM	KM-AUS	KM-EIN	DIG-EIN X	DIG-AUS X
URGELOE	URL-IA	STP-VER	ANL-ABB	UA-PG	UA-SYS	UA-PRFE	UA-SCH
DX0-FE	FE-22	MOF-FE	RAM-FE	DB0-FE	DB1-FE	DB2-FE	KOR-FE
N A U	P E U	B A U	STUE-FE	Z Y K	Q V Z	A D F	WECK-FE
B C F	FE-6	FE-5	FE-4	FE-3	L Z F X	REG-FE	DOPP-FE

In den Steuerbits ist der aktuelle Betriebszustand der CPU vermerkt (>>STP<<), und es sind bestimmte Eigenschaften der CPU markiert (OB 1 geladen, Einprozessorbetrieb, 16KW-Anwenderspeicher usw.). In der obersten Zeile ist als Ursache für den Stoppzustand **STP-BEF** angekreuzt. Es wird angenommen, daß in Ihrem STEP-5-Anwenderprogramm kein STP-Befehl programmiert ist. Daher kommt nur noch ein Stoppbefehl vom Systemprogramm aufgrund eines nicht geladenen Fehler-OBs in Frage. In der untersten Zeile ist die Kennung **LZF** markiert.

Möglicherweise hat das Systemprogramm bei Auftreten eines Laufzeitfehlers festgestellt, daß der zugehörige Fehler-Organisationsbaustein nicht programmiert ist. Da es jedoch verschiedene Laufzeitfehler gibt, und Sie nicht wissen, welcher davon aufgetreten ist, sind die Informationen aus den Steuerbits noch nicht ausreichend.

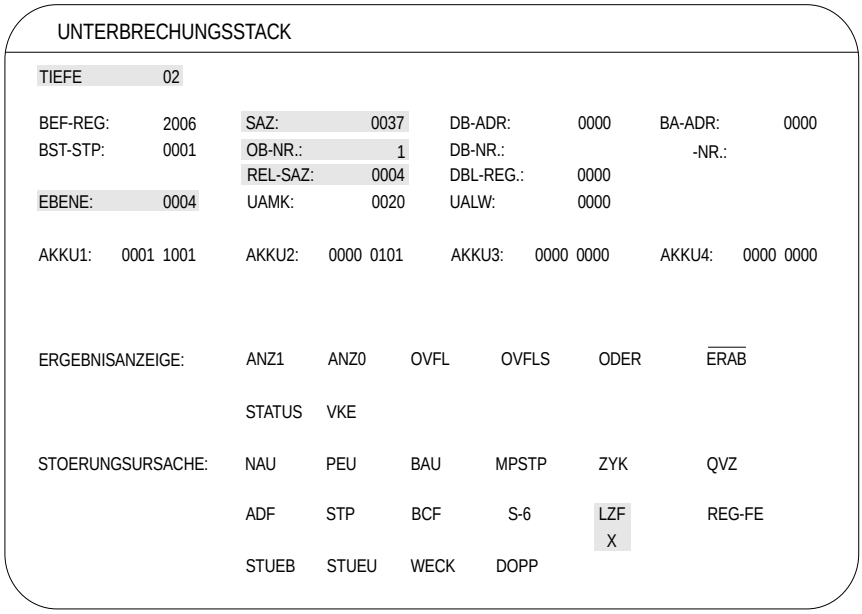
Lassen Sie sich deshalb den USTACK ausgeben:



Der USTACK mit der **Tiefe 01** repräsentiert diejenige Programmbearbeitungsebene, die als letzte vor dem Übergang in den Stoppzustand aktiviert war. An der Kennung **003A** (hinter EBENE) sehen Sie, daß dies der USTACK der Programmbearbeitungsebene **LAUFZEITFEHLER** ist. Im **AKKU 1** ist die Fehlerkennung **00001A01** hinterlegt. Damit wissen Sie, daß es zu einem Laufzeitfehler kam aufgrund eines nicht geladenen Datenbausteins beim Befehl 'A DB'. Da der zugehörige Fehler-OB 19 in unserem Anwenderprogramm nicht vorhanden ist, hat das Systemprogramm die Programmbearbeitung abgebrochen (STP). Im Unterbrechungsanzeigenmaskenwort **UAMK** sind die Unterbrechungsursachen mitgeführt: Die Kennung **0120** entspricht dem Bitmuster **"0000 0001 0010 0000"**. Bit 2^5 (LZF) und Bit 2^8 (STP) sind gesetzt.

Jetzt müssen Sie noch herausfinden, in welchem Baustein und durch welchen Befehl der Laufzeitfehler verursacht worden ist.

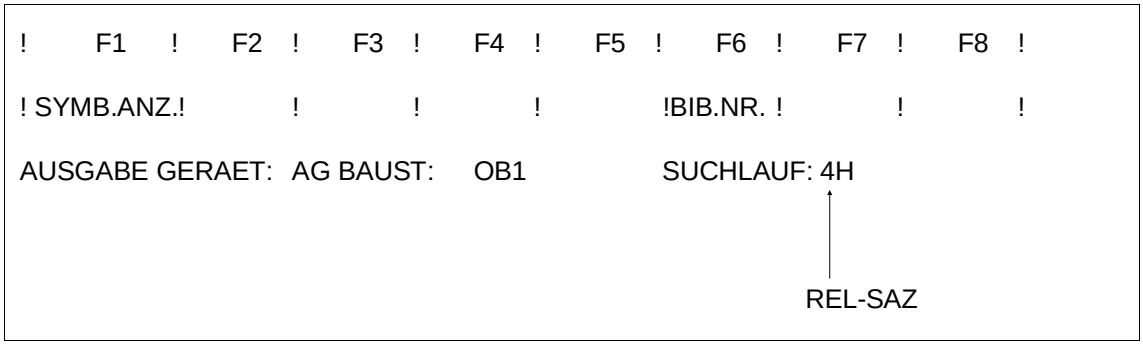
Durch Weiterschalten erscheint am PG der USTACK mit der **Tiefe 02**:



An der Kennung **0004** (hinter EBENE) sehen Sie, daß dies der USTACK der unterbrochenen Programmbearbeitungsebene **ZYKLUS** ist. Der STEP-Adreßzähler (**SAZ)** zeigt auf die Adresse **0037H**. Auf dieser Absolutadresse ist der fehlerverursachende Befehl im Anwenderspeicher hinterlegt. Sein Code ist mit **2006 (BEF-REG)** angegeben. Mit Hilfe der Liste "Auflistung des Maschinencodes" in der Operationsliste kann er als STEP-5-Operation **'ADB 6'** entschlüsselt werden.

Die Unterbrechung ist im Organisationsbaustein **OB 1** aufgetreten. Innerhalb des OB 1 liegt der fehlerverursachende Befehl auf der Relativadresse **0004 (REL-SAZ)**. Dieser Befehl führte, wie Sie bereits festgestellt haben, zu einem Laufzeitfehler (siehe UAMK, Bit 2^5, und STOERUNGSURSACHE).

Lassen Sie sich jetzt am Programmiergerät über die Online-Funktion "Ausgabe Baustein/SUCHLAUF" den fehlerhaften Befehl ausgeben. Geben Sie dazu den betreffenden Baustein (OB 1) und die Relativadresse des Befehls ein.



Am PG sehen Sie nach erfolgtem Suchlauf den Befehl **"A DB 6"**, der für die Unterbrechung verantwortlich ist, da ein Datenbaustein mit der Nummer 6 im Anwenderspeicher nicht vorhanden ist.

```
OB 1

NETZWERK 1    0000
0004      :A  DB 6       fehlerverursachender Befehl
0005      :
0006      :
0007      :
0008      :BE
```

Literaturhinweise 13

Literaturhinweise

/1/ S5-135U/155U
CPU 922/CPU 928/CPU 928B/CPU 948
Tabellenheft

Bestell-Nr. 6ES5 997-3UA12

/2/ Systemhandbuch S5-135U/155U

Bestell-Nr. 6ES5 998-0SH11

/3/ Handbuch STEP 5

Bestell-Nr. C79000-G8500-C140

/4/ GRAPH 5: Ablaufsteuerungen graphisch programmieren unter dem Betriebssystem S5-DOS SIMATIC S5

Bestell-Nr. 6ES5 998-1SA01

/5/ Standard-Funktionsbausteine
Hantierungsbausteine CPU 922, CPU 922, CPU 928B
Automatisierungsgerät S5-135U, S5-155U

/6/ SINEC
Handbuch
CP 143 mit COM 143

Bestell-Nr. 6GK1970-1AB43-0AA0

/7/ Hans Berger:
Automatisieren mit SIMATIC S5-155U

SIEMENS AG ISBN 3-8009-1538-3

/8/ Speicherprogrammierbare Steuerungen
Grundbegriffe

SIEMENS AG
Bestell-Nr. E80850-C293-X-A2

/9/ Katalog ST 59: Programmiergeräte
SIMATIC S5

/10/ Katalog ST 54.1: Automatisierungsgeräte
S5-135U, S5-155U und S5-155H

/11/ Katalog ST 57: Standard-Funktionsbausteine
und Treiberprogramme für
Automatisierungsgeräte der U-Reihe
SIMATIC S5

/12/ Handbuch SCL

Bestell-Nr. C79000-G8500-C162

/13/ Reglerstruktur R64

/14/ AG S5-135U
Kommunikation CPU 928B

Bestell-Nr. 6ES5 998-0CN11

Eine Einführung in das Themengebiet "Speicherprogrammierbare Steuerungen (SPS)" finden Sie in dem Buch:

SPS – Speicherprogrammierbare Steuerungen vom Relaisersatz zum CIM-Verbund: Einführung und Übersicht

Oldenbourg Verlag, ISBN 3-486-21114-5

Verzeichnisse

14

Inhalt von Kapitel 14

Abkürzungsverzeichnis

Abkürzungen

(Die Erklärung der speziellen Abkürzungen des USTACK finden Sie in Abschnitt 5.4)

ADF	Adressierfehler
AG	Automatisierungsgerät
AKKU-1 (2, 3, 4)-L	Low-Wort im Akkumulator 1 (2, 3, 4), 16 bit
AKKU-1 (2, 3, 4)-H	High-Wort im Akkumulator 1 (2, 3, 4), 16 bit
AKKU-1 (2 ,3, 4)-LL	Low-Byte des Low-Wort im Akkumulator 1 (2, 3, 4), 8 bit
AKKU-1 (2, 3, 4)-LH	High-Byte des Low-Wort im Akkumulator 1 (2, 3, 4), 8 bit
ANZ 1, ANZ 0	Wort-Anzeigen, codiert ("Flags")
AWL	Anweisungsliste
ANZW	Anzeigenwort
BASP	Befehlsausgabe sperren (Signal an S5-Bus)
BCD	Binär codierte Dezimalzahl
BR	Basisadreßregister
BSTACK	Bausteinstack
CP	Kommunikationsprozessor
CPU	central processing unit, Zentralprozessor, Zentralbaugruppe (ZBG)
DB	Datenbaustein
DBA	Datenbaustein-Anfangsadresse (im Register 6)
DBL	Datenbaustein-Länge (im Register 8)
DX	Erweiterter Datenbaustein
EG	Erweiterungsgerät
EPROM	erasable progammable read only memory (UV-löschbarer, programmierbarer Nur-Lese-Speicher)
$\overline{\text{ERAB}}$	Erstabfrage (Bit-Anzeige)
FB	Funktionsbaustein
FUP	Funktionsplan
FX	Erweiterter Funktionsbaustein
IM	Interface Modul (Schnittstellenmodul)
IP	Intelligente Peripheriebaugruppe

KOP	Kontaktplan
KOR	Koordinatorbaugruppe
LED	light-emitting diode (Anzeige-Element)
NAU	Netzausfall
OB	Organisationsbaustein
OR	Oder (Bit-Anzeige)
OS	Overflow speichernd (Wortanzeige)
OV	Overflow (Wortanzeige)
PA	Prozeßabbild
PAA	Prozeßabbild der Ausgänge
PAE	Prozeßabbild der Eingänge
PAFE	Parametrierfehler-Byte
PARE	Parityfehler
PB	Programmbaustein
PEU	Ausfall der Versorungsspannung vom Erweiterungsgerät
PG	Programmiergerät
QVZ	Quittungsverzug
RAM	random-accesss memory (Schreib-Lese-Speicher)
SAZ	STEP-Adreßzähler
SB	Schrittbaustein
SPU	Betriebssystemprozessor
STA	Status (Bitanzeige)
STS	Stoppanweisung
SUF	Substitutionsfehler
STUEB	BSTACK-Überlauf
STUEU	USTACK-Überlauf
TLAF	Transfer- oder Ladefehler
USTACK	Unterbrechungsstack
VKE	Verknüpfungsergebnis
ZG	Zentralgerät
ZYK	Zyklusfehler

Stichwortverzeichnis

Verzeichnis der Tabellen und Bilder

Verzeichnis der Tabellen

Verzeichnis der Bilder

Anmerkungen/Vorschläge

Ihre Anmerkungen und Vorschläge helfen uns, die Qualität und Benutzbarkeit unserer Dokumentation zu verbessern. Bitte füllen Sie diesen Fragebogen bei der nächsten Gelegenheit aus und senden Sie ihn an Siemens zurück.

Vergessen Sie dabei nicht, Titel und Bestellnummer mit Ausgabestand anzugeben.

Titel Ihres Handbuchs: .....

Bestell-Nr. Ihres Handbuchs: ..... Ausgabestand: .....

Geben Sie bitte bei den folgenden Fragen Ihre persönliche Bewertung mit Werten

von 1 ≙ gut bis 5 ≙ schlecht an.

1. Entspricht der Inhalt Ihren Anforderungen? ☐
2. Sind die benötigten Informationen leicht zu finden? ☐
3. Sind die Texte leicht verständlich? ☐
4. Entspricht der Grad der technischen Einzelheiten Ihren Anforderungen? ☐
5. Wie bewerten Sie die Qualität der Abbildungen/Tabellen? ☐

Falls Sie auf konkrete Probleme gestoßen sind, erläutern Sie diese bitte in den folgenden Zeilen:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

An
Siemens AG
AUT E 1163
Östl. Rheinbrückenstraße 50

76181 Karlsruhe

–

Absender:
Ihr Name: . . . . . .
Ihre Funktion: . . . . . .
Ihre Firma: . . . . . .
Straße: . . . . . .
PLZ, Ort: . . . . . .
Telefon: . . . . . .

Bitte kreuzen Sie Ihren zutreffenden Industriezweig an:

☐ Automobilindustrie
☐ Chemische Industrie
☐ Elektroindustrie
☐ Nahrungsmittel
☐ Leittechnik
☐ Maschinenbau
☐ Pharmazeutische Industrie
☐ Kunststoffverarbeitung
☐ Papierindustrie
☐ Textilindustrie
☐ Transportwesen
☐ Petrochemie

Andere . . . . . .