Logicmaster 90-70 Programmiersoftware Referenzhandbuch

Logicmaster 90-70 Programmiersoftware Referenzhandbuch

GE Fanuc Logicmaster TM 90-70 Programmiersoftware

Referenzhandbuch

SPEICHER-

PROGRAMMIERBARE

STEUERUNGEN

GFK-0265D-GE

GE Fanuc Automation

DIE BEGRIFFE VORSICHT, ACHTUNG UND HINWEIS, WIE SIE IN DIESER PUBLIKATION VERWENDET WERDEN.

VORSICHT

In dieser Veröffentlichung werden **VORSICHT**-Hinweise verwendet, um darauf hinzuweisen, daß innerhalb der beschriebenen Geräte gefährliche Spannungen, Ströme, Temperaturen oder andere Bedingungen, die körperliche Schäden hervorrufen können, vorkommen.

Wo Unaufmerksamkeit entweder körperliche Schäden oder eine Beschädigung des Gerätes verursachen könnte, werden **VORSICHT**-Hinweise verwendet.

ACHTUNG

ACHTUNG-Hinweise werden dort verwendet, wo das Gerät bei unsachgemäßer Vorgehensweise beschädigt werden könnte.

HINWEIS

HINWEISE sollen nur die Aufmerksamkeit des Lesers auf Informationen lenken, die besonders wichtig für das Verständnis und die Bedienung des Gerätes sind.

Dieses Dokument stützt sich auf Informationen, die zum Zeitpunkt seiner Veröffentlichung verfügbar waren. Obwohl alle Anstrengungen unternommen wurden, den Inhalt so genau wie möglich zu gestalten, können die hier enthaltenen Informationen nicht den Anspruch erheben, alle Details oder Veränderungen von Software und Hardware abzudecken, oder jede Möglichkeit im Zusammenhang mit Installation, Betrieb oder Wartung zu berücksichtigen. In diesem Dokument können Merkmale beschrieben sein, die nicht in allen Hard- und Softwaresystemen vorhanden sind. Weder General Electric Company noch GE Fanuc Automation übernehmen eine Verpflichtung, Besitzer dieses Dokumentes über nachträglich durchgeführte Änderungen zu informieren.

Weder General Electric Company noch GE Fanuc Automation übernehmen Verantwortung für die Genauigkeit, Vollständigkeit oder Nützlichkeit der in diesem Dokument enthaltenen Informationen.

Bei den folgenden Bezeichnungen handelt es sich um Warenzeichen für Produkte von GE Fanuc Automation North America, Inc.

Access 90	CIMSTAR	Helpmate	PROMACRO	Serie Sechs
Alarm Master	GEnet	Logicmaster	Serie Eins	Serie 90
CIMPLICITY	Genius	Modelmaster	Serie Drei	VuMaster
CIMPLICITY PowerTRAC	Genius PowerTRAC	ProLoop	Serie Fünf	Workmaster

In diesem Handbuch werden die bei der Programmierung der SPS Serie 90-70 verwendeten Programmanweisungen beschrieben.

Inhalt dieses Handbuches

Die Kapitel dieses Buches entsprechen den Haupt-Programmfunktionsgruppen. Kapitel 1 gibt einen Überblick über den Inhalt dieses Referenzhandbuches. Lesen Sie daher zuerst Kapitel 1 und schlagen dann bei Bedarf die einzelnen Funktionen nach.

Dieses Handbuch umfaßt die folgenden Kapitel und Anhänge:

Kapitel 1. Anfangen: In diesem Kapitel wird erläutert, wie Sie die einzelnen Themenkreise in diesem Handbuch finden können.

Kapitel 2. Programmorganisation und Anwenderreferenzen/-daten: Kapitel 2 informiert Sie über Programmstruktur, Programmdaten und Anwenderreferenzen.

Kapitel 3. Relaisfunktionen: Dieses Kapitel beschreibt Spulen und Kontakte.

Kapitel 4. Zeitglieder und Zähler: Der Speicherbedarf von Zeitgliedern und Zählern sowie der Einsatz dieser Funktionen werden hier beschrieben.

Kapitel 5. Arithmetische Funktionen: Beschreibung der mathematischen Anweisungen.

Kapitel 6. Relationale Funktionen: Die Verwendung von Funktionen, mit denen zwei Datenwerte verglichen werden, wird hier beschrieben.

Kapitel 7. Bitoperationsfunktionen: Dieses Kapitel beschreibt die Funktionen, die zur Durchführung logischer Operationen bzw. zum Verschieben oder Drehen von Bitfolgen erforderlich sind.

Kapitel 8. Datenverschiebefunktionen: Hier werden Programmanweisungen zur Manipulation von Datenblöcken beschrieben. Hierzu gehören auch die Anweisungen, die benötigt werden, wenn das SPS-System VME-Module am VME-Bus enthält.

Kapitel 9. Datentabellenfunktionen: In diesem Kapitel wird die Verwendung von Programmieranweisungen erläutert, mit denen Daten aus Tabellen im Speicher gelesen oder dorthin geschrieben werden können.

Kapitel 10. Konvertierungsfunktionen: In diesem Kapitel wird die Verwendung von Funktionen erläutert, mit denen ein Datentyp in einen anderen verwandelt werden kann.

Kapitel 11. Steuerfunktionen: Die Anweisungen zur Veränderung des Programmablaufs, zur Durchführung von E/A-Aktualisierung und zum Einfügen erläuternden Texts in ein Programm werden hier beschrieben. Dieses Kapitel beschreibt auch, wie ein spezieller Systemservice (z.B. Daten von der CPU lesen) angefordert werden kann. Ebenfalls wird der Ablauf des CPU-Zyklus hier beschrieben.

Anhang A. Glossar: Dieser Anhang enthält ein Verzeichnis der auf die SPS Serie 90-70 bezogenen Hard- und Softwareausdrücke.

Anhang B. CPU-Zyklus: In diesem Anhang wird der Ablauf des CPU-Zyklus beschrieben.

Anhang C. Programmierung mnemonischer Anweisungen: Zusammenfassung der mnemonischen Anweisungen, die eingegeben werden können, während ein Suchlauf abläuft oder das Programm editiert wird.

Anhang D. Speicherbelegung: Dieser Anhang enthält ein Arbeitsblatt, mit dem Sie die Gesamtanzahl der bereits belegten bzw. noch freien Bytes im Speicher (%R, %AI, %AQ, %P, %L) ermitteln können.

Anhang E. Tastenfunktionen: Liste der speziellen Tastenbelegungen für die Logicmaster 90-70 Software.

Zugehörige Veröffentlichungen

- GFK-0255 - *Serie 90™ Programmierbares Coprozessormodul, Handbuch*
- GFK-0262 - *Serie 90™-70 speicherprogrammierbare Steuerung, Installations- und Betriebshandbuch*
- GFK-0263 - *Logicmaster™ 90-70 Programmiersoftware, Anwenderhandbuch*
- GFK-0350 - *Logicmaster™ 90-70, wichtige Produktinformationen*
- GFK-0398 - *Serie 90™-70, Buscontroller-Handbuch*
- GFK-0448 - *Richtlinien für die Auswahl von VME-Modulen anderer Hersteller*
- GFK-0533 - *GEnet Fertigungs-LAN für MAP 3.0 und MMS-Ethernet der Serie 90-70 SPS, Anwenderhandbuch*

Kapitel 1

Kapitel 2

KAPITEL 1

In diesem Handbuch werden die Programmieranweisungen beschrieben, die bei der Erstellung von Kontaktplanprogrammen für die speicherprogrammierbaren Steuerungen Serie 90-70 benutzt werden können. Im *Anwenderhandbuch zur Logicmaster 90-70 Programmiersoftware* (GFK-0263) wird beschrieben, wie Sie das Logicmaster 90-70 Softwarepaket installieren müssen und wie Sie Programme erstellen, übertragen, editieren und ausdrucken können.

Bei einer effizienten Programmierung kann die SPS-Konfiguration zuerst mit der Konfigurationssoftware abgeschlossen werden. Wurde dieser Schritt noch nicht durchgeführt, dann können Sie anhand der Erläuterungen des Anwenderhandbuchs GFK-0263 entscheiden, ob es sinnvoll ist, mit der Programmierung zu beginnen.

Struktur dieses Handbuches

In diesem Kapitel wird erläutert, wie Sie in diesem Handbuch die gewünschten Informationen finden können.

Programmorganisation, Anwenderreferenzen, Datentypen: siehe Kapitel 2

Kapitel 2 beschreibt die Programmstrukturen und erläutert, wie den Programmfunktionen Referenzadressen zugewiesen werden.

Kontakte, Spulen und Verbindungen: siehe Kapitel 3

Die elementarsten Bestandteile eines Programms sind die Relaisfunktionen, die in Kapitel 3 beschrieben werden. Diese Kontakte und Spulen, die zur Steuerung des logischen Programmablaufs verwendet werden könne, stellen Geräteein- und -ausgänge dar. Diese Funktionen geben die Ausführung anderer Programmfunktionen in einem Strompfad frei oder verhindern sie und zeigen Ausgangszustände an. Die Logicmaster 90-70 Software umfaßt zahlreiche Kontakt- und Spulentypen, die eine maximale Flexibilität bei der Programmierung gewährleisten.

Zeitglieder und Zähler: siehe Kapitel 4

Kapitel 4 informiert Sie über Zeitglieder zur Einschaltverzögerung und Zeiterfassung sowie über Auf- und Abwärtszähler.

Arithmetische Funktionen: siehe Kapitel 5

In Kapitel 5 werden die Funktionen zur Durchführung von Additionen, Subtraktionen, Multiplikationen, Divisionen und Modulo-Divisionen erläutert. (Division und Modulo-Division sind ähnliche Funktionen mit unterschiedlichen Ausgangswerten: eine Division ergibt einen Quotienten, während eine Modulo-Division einen Rest ergibt).

Jede arithmetische Funktion verwendet zwei ganzzahlige oder reelle Variablen mit oder ohne Vorzeichen oder doppelter Genauigkeit, die vom gleichen Typ sind. Unterscheiden sich die Typen der verwendeten Zahlen (soll z.B. eine vorzeichenbehaftete ganze Zahl mit einer 4-stelligen BCD-Zahl kombiniert werden), dann müssen Sie zuerst eine Konvertierungsfunktion (siehe Kapitel 10) programmieren, mit der die Eingangswerte aufeinander angepaßt werden.

Konvertierung von Datentypen: siehe Kapitel 10

Zahlreiche Programmfunktionen (z.B. die arithmetischen Funktionen) benötigen gleiche Datentypen. Mit den Konvertierungsfunktionen können Sie eine Zahl in ein Wort oder doppeltlanges Wort, einen BCD-Wert oder in einen ganzzahligen oder reellen Wert mit oder ohne Vorzeichen oder mit doppelter Genauigkeit umwandeln.

Vergleich zweier Zahlen: siehe Kapitel 6

Mit den in Kapitel 6 beschriebenen Vergleichsfunktionen können Sie feststellen, ob zwei Zahlenwerte (die vom gleichen Typ sein müssen) gleich groß sind oder ob einer der Werte größer oder kleiner als der andere ist.

Arbeiten mit Bitfolgen: siehe Kapitel 7

Kapitel 7 erläutert, wie Datenverschiebungen und Boolsche Operationen auf Bitfolgen angewandt werden können:

1. Anwendung von Boolschen Operationen (AND, OR, XOR, NOT) auf zwei Bitfolgen gleicher Länge.

2. Erstellen eines Ausgangsstrings, der die Kopie eines Eingangsstrings darstellt, bei dem jedoch die Bits negiert, verschoben oder gedreht sind. Mit der Datenverschiebefunktion kann auch ein Speicherbereich gelöscht oder mit Daten gefüllt werden.

3. Ein Bit in einem String finden und seinen Zustand feststellen oder das Bit auf 0 oder 1 setzen. Verwenden Sie eine der Datenverschiebefunktionen, um größere Speicherbereiche zu löschen.

4. Vergleich zweier Bitstrings.

Mit den in Kapitel 8 beschriebenen Datenverschiebefunktionen können Sie ähnliche Funktionen des gleichen Typs durchführen. Verwenden Sie die in Kapitel 9 beschriebenen Datentabellenfunktionen, wenn Sie Daten in und aus Tabellen verschieben wollen, die als FIFO- oder LIFO-Stacks organisiert sind.

Datenverschiebung in Tabellen: Siehe Kapitel 9

Verwenden Sie die Datentabellenfunktionen, wenn Sie Daten in und aus Tabellen verschieben wollen, die als FIFO- (z.B. Teile auf einer Fertigungsstraße) oder LIFO-Stacks (z.B. ein Palettenstapel) organisiert sind.

Datenverschiebung: Siehe Kapitel 8

In Kapitel 8 ist die Erstellung von Programmteilen beschrieben, mit denen Sie:

1. Daten auf einen anderen Platz kopieren können. Die Daten werden dabei als einzelne Bits kopiert.
2. Einen Block mit Konstanten in den Speicher verschieben können.
3. Einen Bereich des bit- oder wortorientierten Speichers löschen können.
4. Daten von einem Speicherbereich zu einem anderen verschieben können, wobei die herausfallenden Daten erfaßt werden.
5. Zwei Bytes innerhalb eines Wortes oder zwei Worte innerhalb eines Doppelwortes gegeneinander vertauschen können (im Bereich eines oder mehrerer Worte/Doppelworte).

E/A-Aktualisierung (DO I/O und SUS I/O): siehe Kapitel 11

Mit den Funktionen DO I/O und SUS I/O können Sie den normale Aktualisierungsablauf bei den E/A-Modulen und Genius-E/A-Modulen des Systems innerhalb eines CPU-Zyklus verändern. Diese Funktionen können nicht für die VME-Module am VME-Bus verwendet werden.

Kommunikation mit VME-Modulen: siehe Kapitel 8

Enthält das System VME-Module am VME-Bus, dann müssen für den gesamten Datenverkehr mit diesen Modulen (einschließlich E/A-Aktualisierung) spezielle Programmanweisungen verwendet werden. Der Datenverkehr zwischen VME-Modulen und der SPS Serie 90 ist nur während des Programmteils des CPU-Zyklus als Antwort auf spezielle Programmanweisungen möglich.

Kommunikation mit anderen Modulen: siehe Kapitel 8

Die Funktion COMMREQ erlaubt den Datenaustausch zwischen dem Programm und einem intelligenten Modul im System (z.B. Datenübertragung zu einem PCM).

Spezielle Dienste von der SPS: siehe Kapitel 11

Mit der in Kapitel 11 beschriebenen Funktion SVCREQ können Sie:

1. Das Zeitglied für konstante Zyklusdauer lesen und einstellen.
2. Windowzustände lesen.
3. Zustand und Werte des Programmiergeräte-Kommunikationswindows einstellen.
4. Zustand und Werte des Systemkommunikations-Windows einstellen.
5. (Reserviert.)
6. (Reserviert.)
7. Zustand und Werte der Echtzeituhr einstellen/lesen.
8. Zeitüberwachung (Watchdog) rücksetzen.
9. Zeit seit Zyklusbeginn lesen.
10. Den Namen des Programms lesen, in dem sich der Block befindet.

11. SPS-Kennung lesen.
12. RUN-Zustand der SPS lesen.
13. SPS abschalten.
14. Fehlertabellen löschen.
15. Den letzten Eintrag in die Fehlertabelle lesen.
16. Betriebszeituhr lesen.
17. Masken für E/A-Interrupts setzen oder löschen.
18. Den E/A-Overridezustand lesen.
19. Ablauf freigeben/sperren.
20. Fehlertabellen lesen.

Strompfadkommentare: siehe Kapitel 11

Mit der Kommentarfunktion können Sie Textkommentare zu den Strompfaden im Programm einfügen.

Programmablauf verändern: siehe Kapitel 11

Mit der MCR-Funktion können Sie Teile des Programms mit negativer Logik ausführen; die SKIP-Funktion erlaubt das Überspringen von Programmteilen.

Einen Prgrammblock aufrufen: siehe Kapitel 11

Mit der CALL-Anweisung können Sie innerhalb des Programms direkt auf den genannten Programmblock springen.

Glossar: siehe Anhang A

In Anhang A finden Sie ein Glossar der im Zusammenhang mit der SPS Serie 90-70 verwendeten Fachausdrücke.

CPU-Zyklus: siehe Anhang B

Anhang B beschreibt den Ablauf des CPU-Zyklus.

Mnemonische Anweisungen: siehe Anhang C

In Anhang C finden Sie eine Liste der mnemonischen Anweisungen, die bei der Logicmaster 90 Software verwendet werden können.

Speicherbelegung: siehe Anhang D

Mit dem Arbeitsblatt in Anhang D können Sie die Gesamtanzahl der bereits belegten Bytes im Speicher sowie den für Anwenderprogramme verfügbaren Speicherplatz ermitteln.

Tastenfunktionen: siehe Anhang E

In Anhang E finden Sie eine Liste der in der Logicmaster 90 Software verwendeten Tastenbelegungen.

KAPITEL 2

Dieses Kapitel enthält allgemeine Hinweise, mit denen ein grundsätzliches Verständnis für die Erstellung von Kontaktplanprogrammen vermittelt werden soll.

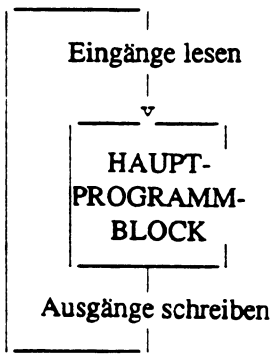
Inhalt von Kapitel 2

Kapitel 2 enthält folgende Abschnitte:

Abschnitt 1. Programmstruktur: Dieser Abschnitt erläutert die Programmblöcke und ihr Zusammenwirken.

Abschnitt 2. Anwenderreferenzen: In diesem Abschnitt werden für den Anwender zugängliche diskrete und Registerrefenzen, indirekte Referenzen und Datentypen beschrieben. Dieser Abschnitt enthält auch Informationen über konfigurierbare und nicht konfigurierbare Fehlerreferenzen, Fehlersuchreferenzen, Fehlerkontakte, Punktfehler und Alarmkontakte.

Abschnitt 3. Funktionsblock-Struktur: Dieser Abschnitt beschreibt den Daten- und Stromfluß durch eine Funktion im Kontaktplanprogramm.



Abschnitt 1 - Programmstruktur

Das Anwenderprogramm enthält Logik, die beim Start verwendet wird und die von der SPS zyklisch durchlaufen wird. Eine Liste der Programmgrößen und Referenzgrenzen der einzelnen CPU-Modelle finden Sie in GFK-0262.

Hauptprogrammblock

Jedes Programm für die SPS Serie 90-70 besteht aus einer oder mehreren Einheiten, die Programmblock genannt werden. Jeder Programmblock kann bis zu 16 k Worte enthalten.

Es gibt immer einen Hauptprogrammblock, der die Logik enthält, die bei jedem Zyklus ausgeführt wird.

Programmblöcke

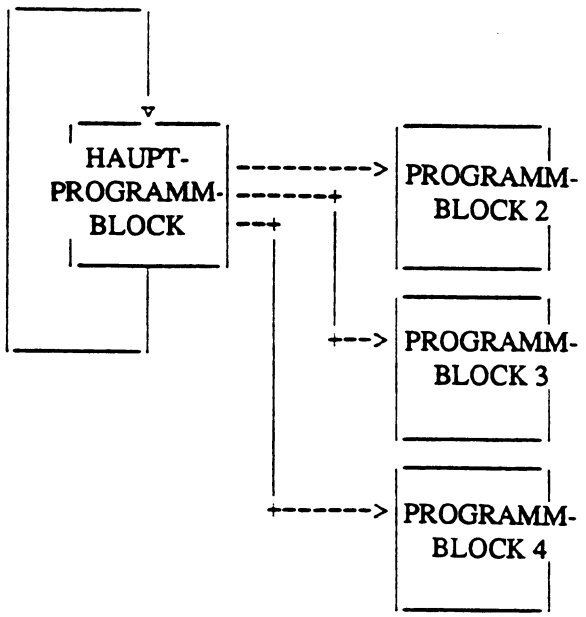
Während der Programmbearbeitung kann ein Hauptprogrammblock andere Programmblöcke aufrufen. Jeder Programmblock kann dabei ebenfalls 16 k Worte umfassen. Die Verwendung zusätzlicher Programmblöcke ist wahlweise, das Aufteilen eines Programms in kleinere Programmblöcke kann jedoch die Programmierung vereinfachen und den gesamten Programmumfang reduzieren.

HINWEIS

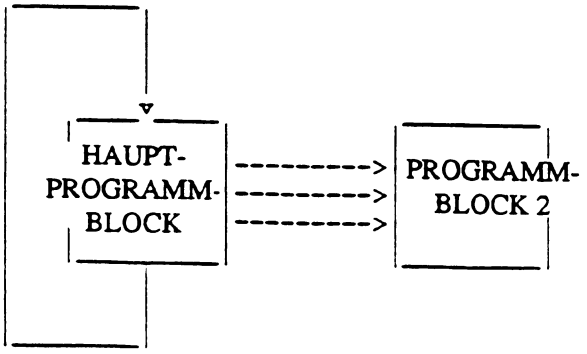
Maximal können in einem Programmblock 64 Programmblock-Aufrufe erfolgen.

Beispiele für die Verwendung von Programmblöcken

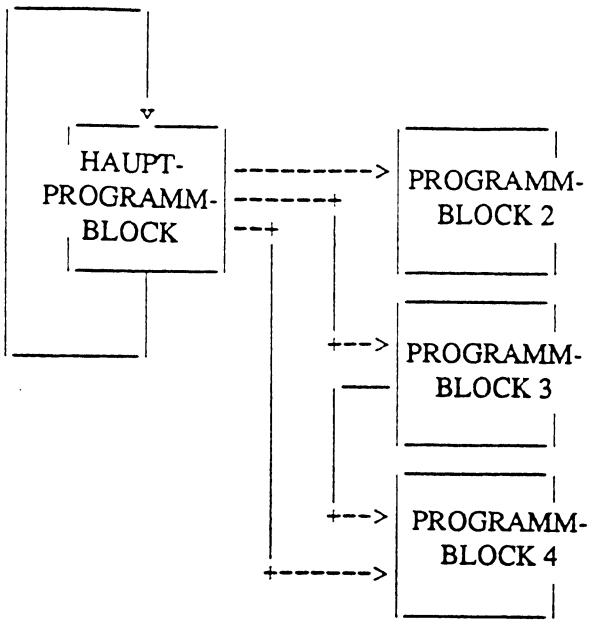
Die Logik eines Programms kann zum Beispiel in drei Programmblöcke unterteilt werden, die jeweils bei Bedarf von einem Hauptprogrammblock aufgerufen werden können. Ein Programmblock kann jedoch nicht den Hauptprogrammblock aufrufen. In diesem Beispiel kann der Hauptprogrammblock nur gerade soviel Logik enthalten, wie zur Steuerung der übrigen Programmblöcke erforderlich ist.



Ein Programmblock kann während der Programmbearbeitung beliebig oft aufgerufen werden. Ein Programmteil, der innerhalb eines Programms wiederholt benötigt wird, kann als Programmblock ausgeführt werden, der dann bei Bedarf aufgerufen wird. Auf diese Weise wird die Länge des gesamten Programms reduziert.



Programmblöcke können auch von anderen Programmblöcken aus aufgerufen werden. Ein Programmblock kann sich auch selbst aufrufen.



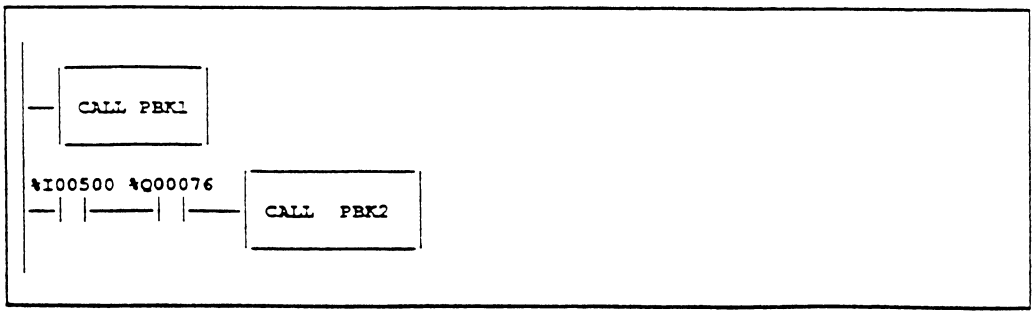
Bei der Logicmaster 90-70 Software ist die Anzahl der möglichen Aufrufebenen theoretisch unbegrenzt. Bei der SPS Serie 90-70 wird jedoch ein störungsfreier Betrieb nur bis zu maximal acht Ebenen garantiert. Nachdem die Programmbearbeitung vom Hauptprogrammblock zum ersten Programmblock (Ebene 1) übergegangen ist, können also noch sieben weitere Aufrufe zu anderen Programmblöcken durchgeführt werden, ehe das Programm wieder zum Hauptprogrammblock zurückkehren muß. In der vorstehenden Abbildung werden vier Aufrufebenen dargestellt. Werden mehr als acht Aufrufe durchgeführt, dann entsteht im SPS-Stack ein Überlauf, der die SPS anhält und einen Fehlereintrag erzeugt.

HINWEIS

Ehe ein Programmblock in einem Aufruf oder einer Interruptanweisung verwendet werden kann, muß er in den Programmblockvereinbarungen definiert werden. Informationen zur Programmblockvereinbarung finden Sie in Kapitel 3 von GFK-0263.

Programmaufruf

Ein Programmblock wird bearbeitet, wenn er vom Programm im Hauptprogrammblock oder einem andern Programmblock aufgerufen wird:

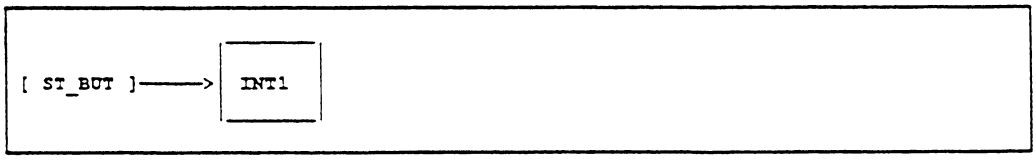


In diesem Beispiel wird PBK1 immer aufgerufen. Der Aufruf des Programmblocks kann über Bedingungslogik gesteuert werden. Damit PBK2 aufgerufen wird müssen sowohl Eingang %I00500 als auch Ausgang %Q00076 wahr sein.

Interruptblöcke

Ein Programmblock kann auch durch den Interrupteingang bestimmter Hardwaremodule angesteuert werden. Bei dem 32-Punkt-Eingangsmodul für 24 V= (IC697MDL650) kann zum Beispiel der erste Eingang als Interrupteingang konfiguriert werden. In diesem Fall kann dieser Eingang als Auslöser dienen, der die Ausführung eines Programmblocks anstößt.

Die von einem Interrupt ausgelöste Bearbeitung eines Blocks wird anstelle der Bearbeitung des Hauptprogramms durchgeführt. Der Ablauf des Hauptprogramms wird unterbrochen und erst wieder aufgenommen, nachdem der Interruptblock vollständig bearbeitet wurde.

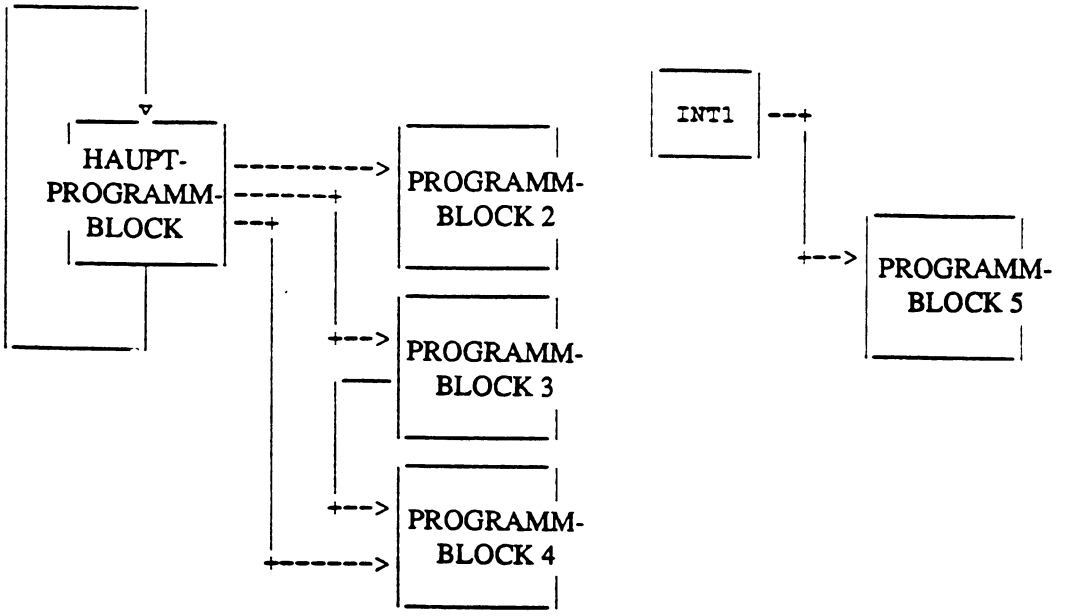


In der vorstehenden Abbildung ruft der Auslöser ST_BUT den Programmblock INT1 auf, wenn die Eingabe von einem an Eingang 1 angeschlossenen Stopschalter auf "1" geht. Der Interrupt kann bei ansteigender oder abfallender Flanke oder bei einem Zustandswechsel erzeugt werden.

HINWEIS

Das Logicmaster 90-70 Softwarepaket stellt zeitgesteuerte Interrupts zur Verfügung, die jedoch von der Firmware der 90-70 mit Ausgabestand 3 nicht unterstützt werden.

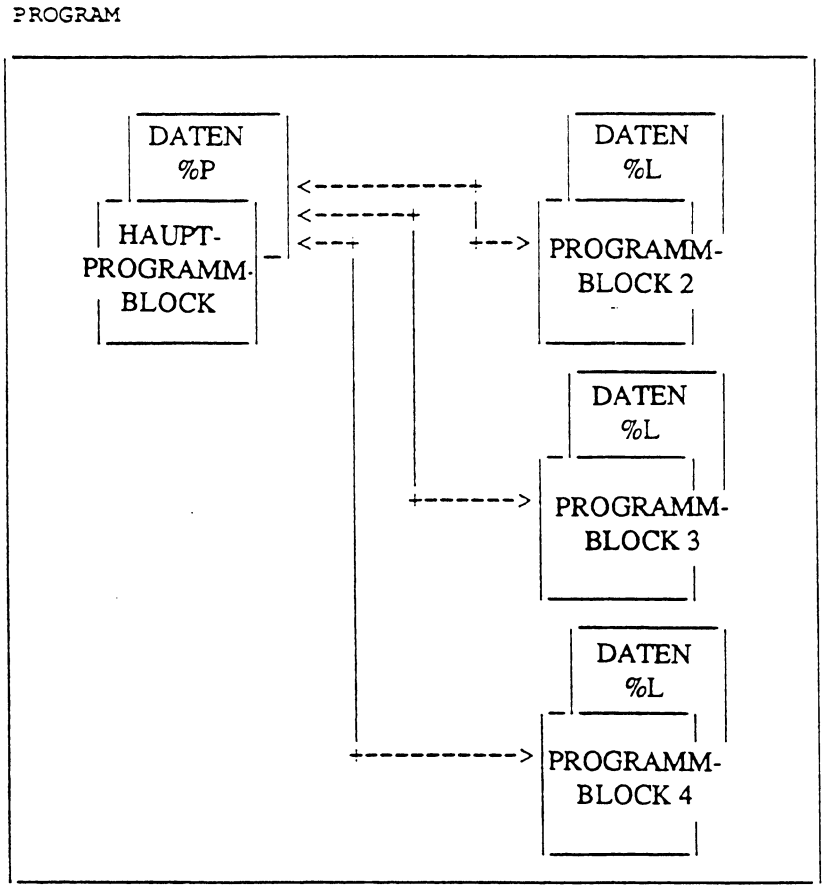
Damit der Programmblock auf diese Weise eingesetzt werden kann, muß er zunächst in den Programmvereinbarungsteil aufgenommen und über eine Interruptvereinbarung mit dem Interrupt verknüpft werden. Der gleiche Programmblock kann auch auf normale Weise im Programm aufgerufen werden.



Programmblöcke und Daten

Zu jedem Programmblock gehört ein Datenblock. Der Haupt-Datenblock wird mit %P angesprochen, alle anderen Datenblöcke mit %L.

Die Größe des Datenblocks hängt von der höchsten %L-Referenz im zugehörigen Programmblock und von der höchsten %P-Referenz in allen Programmblöcken ab.



Sämtliche Programmblöcke können Daten verwenden, die mit dem Hauptprogrammblock verknüpft sind (%P). Programmblöcke können ihre eigenen %L-Referenzen ebenso verwenden wie die %P-Referenzen, die allen Blöcken zur Verfügung stehen. Der Hauptprogrammblock kann %L nicht verwenden.

Abschnitt 2 - Anwenderreferenzen und Daten

Die in einem Anwenderprogramm verwendeten Daten werden entweder als diskrete Referenzen oder als Registerreferenzen gespeichert.

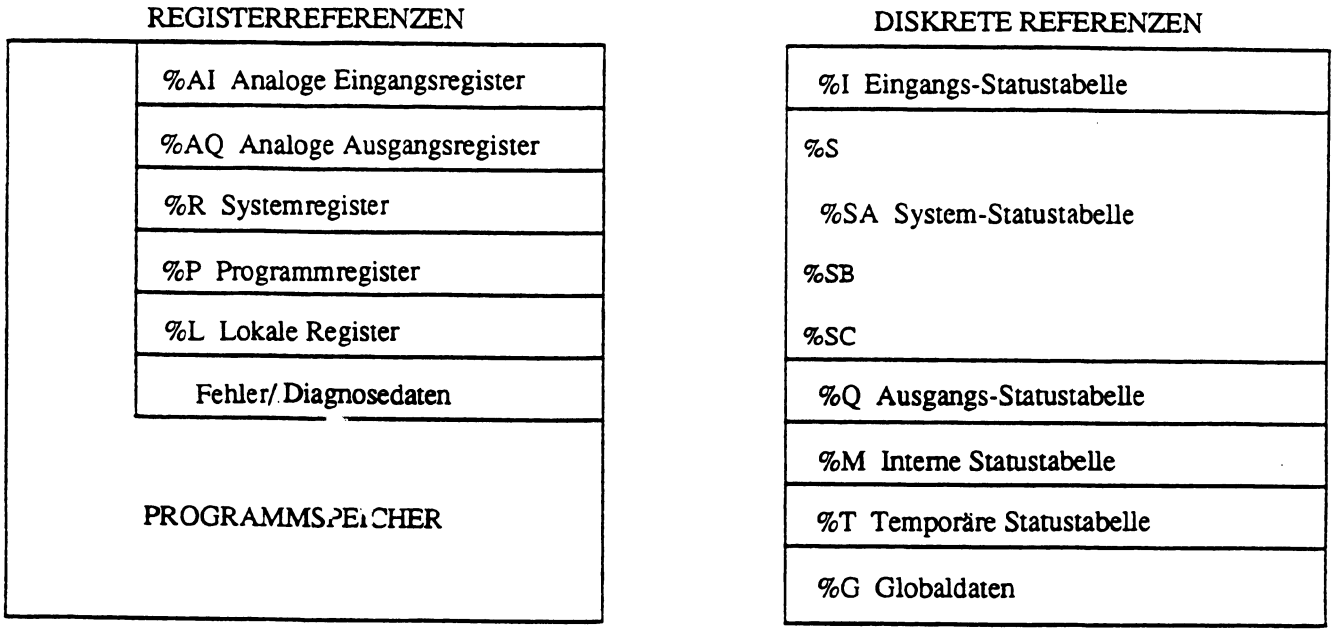


Tabelle 2-1. Registerreferenzen

Typ	Beschreibung
%R	Mit dem Präfix %R werden System-Registerreferenzen zugewiesen, in die Programmdaten abgespeichert werden können (z.B. Berechnungsergebnisse).
%AI	Das Präfix %AI steht für ein analoges Eingangsregister. Ihm folgt die Registeradresse der Referenz (z.B. %AI0015). In einem analogen Eingangsregister steht der Wert eines Analogeingangs oder ein anderer Wert.
%AQ	Das Präfix %AQ steht für ein analoges Ausgangsregister. Ihm folgt die Registeradresse der Referenz (z.B. %AQ0056). In einem analogen Ausgangsregister steht der Wert eines Analogausgangs oder ein anderer Wert.
%P	Mit dem Präfix %P werden Programmregisterreferenzen zugewiesen, in denen Programmdaten aus dem Hauptprogrammblock gespeichert werden. Auf diese Daten kann von allen Programmblöcken aus zugegriffen werden. Die Länge des %P-Datenblocks hängt von der höchsten bei allen Programmblöcken verwendeten %P-Referenz ab. (Weitere Informationen finden Sie in Anhang D).
%L	Mit dem Präfix %L werden lokale Registerreferenzen zugewiesen, in denen spezifische Programmblockdaten abgelegt werden. Die Länge des %L-Datenblocks hängt von der höchsten bei dem zugehörigen Programmblock verwendeten %L-Referenz ab. (Weitere Informationen finden Sie in Anhang D).

HINWEIS

Sämtliche Registerreferenzen bleiben beim Aus- und Einschalten der CPU-Versorgungsspannung erhalten.

Tabelle 2-2. Diskrete Referenzen

Typ	Beschreibung
%I	Das Präfix %I steht für Eingangsreferenzen. Ihm folgt die Referenzadresse in der Eingangstabelle (z.B. %I00121). Die Referenzen %I liegen in der Eingangs-Statustabelle, in der die Zustände sämtlicher Eingangssignale gespeichert sind, die von den Eingangsmodulen im letzten Eingabezyklus empfangen wurden. Den diskreten Eingangsmodulen wird mit der Konfigurationssoftware eine Referenzadresse zugewiesen. Solange keine Referenzadresse zugewiesen wurde, können vom Modul keine Daten empfangen werden.
%Q	Das Präfix %Q steht für physikalische Ausgangsreferenzen. Bei freigegebener Spulenüberprüfungsfunktion wird %Q auf Mehrfachverwendung überprüft. Dem Präfix folgt die Referenzadresse in der Ausgangstabelle (z.B. %IQ00016). Die Referenzen %Q liegen in der Ausgangs-Zustandstabelle, in der die Zustände der Ausgangsreferenzen so gespeichert sind, wie sie zuletzt vom Anwenderprogramm eingestellt wurden. Die Werte der Ausgangs-Zustandstabelle werden am Ende des Programmzyklus an die Ausgangsmodule übertragen. Den diskreten Ausgangsmodulen wird mit der Konfigurationssoftware eine Referenzadresse zugeordnet. Solange einem Ausgangsmodul keine Referenzadresse zugeordnet wurde, werden zu diesem Modul keine Daten übertragen. Eine bestimmte %Q-Referenz kann remanent oder nicht remanent sein.
%M	Das Präfix %M steht für eine interne Referenz. Bei freigegebener Spulenüberprüfungsfunktion wird %M auf Mehrfachverwendung überprüft. Eine bestimmte %M-Referenz kann remanent oder nicht remanent sein.
%T	Präfix %T steht für eine diskrete temporäre Referenz. Diese Referenzen werden nicht auf Mehrfachverwendung überprüft und können daher im gleichen Programm selbst dann mehrfach eingesetzt werden, wenn die Spulenverwendungs-Überprüfungsfunktion aktiviert ist. %T kann dazu verwendet werden, beim Einsatz der Funktionen CUT/PASTE bzw. WRITE/INCLUDE Spulenverwendungskonflikte zu vermeiden. Da dieser Speicherbereich nur für den temporären Gebrauch vorgesehen ist, kann er nicht mit remanenten Merkern verwendet werden. Der Inhalt geht bei Spannungsausfall und bei RUN-STOP-RUN-Übergängen verloren.
%S	Das Präfix %S steht für Systemreferenzen. Mit diesen Referenzen kann auf spezielle SPS-Daten zugegriffen werden (z.B. Zeitglieder, Zyklusdaten, Fehlerdaten). %S, %SA, %SB und %SC können für alle Kontakte verwendet werden. %SA, %SB und %SC können für remanente Spulen -( )- verwendet werden. %S kann als Wort- oder Bitfolge-Eingangsargument in Funktionen oder Funktionsblöcken verwendet werden. %SA, %SB und %SC können als Wort- oder Bitfolge-Ein- oder Ausgangsargument in Funktionen oder Funktionsblöcken verwendet werden.
%G	Das Präfix %G steht für nahtlose Globalreferenzen. Über diese Referenzen kann auf Daten zugegriffen werden, die von mehreren SPS-System gemeinsam benutzt werden. %G-Referenzen können bei Kontakten und remanenten Spulen verwendet werden, da der %G-Speicher immer remanent ist. %G kann nicht mit nicht remanenten Spulen verwendet werden.

Transitionen und Overrides

Den Anwenderreferenzen %I, %Q, %M, und %G sind Transitions- und Overridebits zugeordnet. Die Referenzen %T, %S, %SA, %SB, %SC besitzen Transitionsbits, jedoch keine Overridebits. Die CPU verwendet Transitionsbits für Zähler, Übergangskontakte und Übergangsmerker. Beachten Sie, daß Zähler eine andere Art Transitionsbits verwenden als dies bei Kontakten und Spulen der Fall ist. Transitionsbits für Zähler werden in der Adreßreferenz abgelegt.

Werden Overridebits gesetzt, dann können die zugehörigen Referenzen nur über das Programmiergerät verändert werden, nicht vom Programm oder Eingabegerät.

.i.Datenremanenz

Daten sind "remanent", wenn sie beim Anhalten der SPS gerettet werden. Die SPS Serie 90-70 rettet Programmlogik, Fehlertabellen und Diagnosedaten, Prüfsummen für Programmblöcke, Überschreibungen (Override) und Setzen von Ausgängen, Wortdaten (%R, %L, %P, %AI, %AQ), Bitdaten (%I, %S, %G, Fehlerbits und reservierte Bits), %Q- und %M-Daten (sofern sie nicht mit nicht-remanenten Merkern verwendet werden), sowie Wortdaten, die in %M und %Q gespeichert sind.

%T- sowie %Q- und %M-Daten (die mit nicht-remanenten Merkern verwendet werden) werden nicht gerettet.

%Q-Referenzen sind nicht remanent (d.h. sie werden beim Einschalten gelöscht, wenn die SPS von STOP auf RUN umschaltet), wenn sie mit nicht remanenten Spulen verwendet werden. Zu den nicht remanenten Spulen gehören Merker -( )-, negierte Merker -(/)-, SET-Merker -(S)- und RESET-Merker -(R)-.

Wird eine %Q-Referenz mit einer remanenten Spule zusammen oder als Ausgang eines Funktionsblocks verwendet, dann wird der Inhalt bei einem Spannungsausfall bzw. bei Übergängen RUN-STOP-RUN erhalten. Zu den remanenten Spulen gehören remanente Merker -(M)-, negierte remanente Merker -(/M)-, remanente SET-Merker -(SM)- und remanente RESET-Merker -(RM)-.

Eine %M-Referenz ist nicht remanent, wenn sie mit einer nicht remanenten Spule zusammen verwendet wird. Wird eine %M-Referenz mit einer remanenten Spule zusammen oder als Ausgang eines Funktionsblocks verwendet, dann wird der Inhalt bei einem Spannungsausfall bzw. beim Wechsel RUN-STOP-RUN erhalten.

Die letzte Verwendung einer %Q- oder %M-Referenz legt ihren Remanenzstatus fest.

Indirekte Referenzen

Alle Registerreferenzen (%R, %AI, %AQ, %P und %L) können indirekt angesprochen werden. Diese Funktion kann dann nützlich sein, wenn Sie mehreren Registern den gleichen Wert hinzufügen wollen, ohne die Kontaktplanlogik im Anwenderprogramm zu wiederholen. Sie kann auch bei Schleifen verwendet werden, bei denen die einzelnen Bits im Register solange um die angegebene Konstante oder den angegebenen Wert erhöht werden, bis ein Maximalwert erreicht ist.

Mit indirekten Referenzen wird eine Speicheradresse angegeben, die den Relativzeiger im gleichen Speichertyp der zu verwendenden Daten enthält. Enthält z.B. %L00001 den Wert 5, dann weist @L00001 die SPS an, die Datenadresse %L00005 zu verwenden. Indirekte Referenzen werden auf die gleiche Weise wie direkte Referenzen eingegeben, es wird lediglich das Zeichen @ anstelle des Zeichens % verwendet.

Geltungsbereich der Daten

Jede Anwenderreferenz besitzt einen "Geltungsbereich". Dieser kann sich auf das gesamte System beziehen, oder die Referenz ist zwar für alle Programmblöcke verfügbar, jedoch nicht für einen Hostcomputer, oder sie ist nur lokal innerhalb eines Programmblocks verwendbar.

Anwenderreferenz	Bereich	Geltungsbereich
%I, %Q, %M, %T, %AI, %AQ, Zeitimpulse, Funktionsreferenz, Fehler, Alarm	System	Von jedem Programmblock oder Hostcomputer
%P	Programm	Von jedem Programmblock
%L	Lokal	Innerhalb eines Programmblocks

In einem Programmblock gelten folgende Regeln:

- %R sollte für ein Register benutzt werden, das gemeinsam mit anderen Programmblöcken verwendet wird, oder auf das über ein CCM-Protokoll zugegriffen wird.
- %P sollte für Programmreferenzen eingesetzt werden, die gemeinsam mit anderen Programmblöcken verwendet werden.
- %L sind lokale Referenzen, mit denen der Gebrauch von Registerdaten auf einen Programmblock beschränkt werden kann. Diese lokalen Referenzen sind in anderen Programmteilen nicht verfügbar.
- Auf %P und %L kann über das CCM-Protokoll nicht zugegriffen werden. Zugriff ist jedoch möglich über das SNP-Protokoll, Ethernet oder MAP.

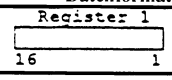
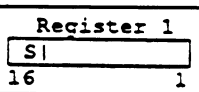
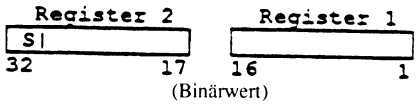
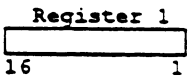
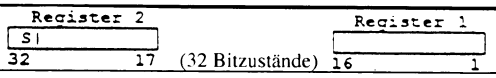
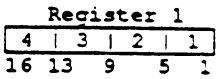
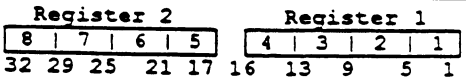
Die Referenzen %I, %Q, %M, %T, %AI und %AQ sind im gesamten System verfügbar.

Anhang D enthält ein Arbeitsblatt, mit dem Sie die Gesamtanzahl der bereits belegten bzw. für das Anwenderprogramm noch freien Bytes im Speicher (%R, %AI, %AQ, %P, %L) ermitteln können.

Datentypen

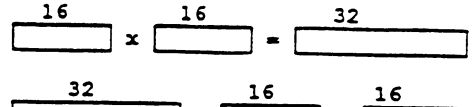
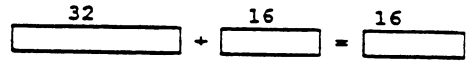
Die folgenden Datentypen sind möglich:

Tabelle 2-3. Datentypen

Typ	Bezeichnung	Beschreibung	Datenformat
UINT	Ganzzahlig ohne Vorzeichen	Dieser Datentyp belegt 16 Datenbits im Speicher. Der zulässige Bereich liegt zwischen 0 und +65.535 (FFFF Hexa).	 (Binärwert)
INT	Vorzeichen-behaftete ganze Zahl	Dieser Datentyp belegt 16 Datenbits im Speicher und wird im Zweierkomplement dargestellt. Der zulässige Bereich liegt zwischen -32.768 und +32.767.	 (Wert im Zweier-komplement)
DINT	Doppelt-genaue ganze Zahl	Dieser Datentyp belegt 32 Datenbits im Speicher (zwei aufeinanderfolgende Speicherplätze) und wird immer mit Vorzeichen dargestellt (Bit 32 ist das Vorzeichenbit). Der zulässige Bereich liegt zwischen -2.147.483.648 und +2.147.483.647.	 (Binärwert)
BIT	Bit	Der Bit-Datentyp ist die kleinste Einheit im Speicher. Ein Bit kann einen von zwei möglichen Zuständen annehmen, 1 oder 0. Eine Bitfolge kann die Länge N haben.	
WORD	Wort	Der Wort-Datentyp verwendet 16 aufeinanderfolgende Bits im Datenspeicher. Diese Bits stellen aber keine Zahl dar, sondern sind unabhängig voneinander. Jedes Bit wird für sich betrachtet und stellt einen eigenen Binärzustand (0 oder 1) dar. Der zulässige Bereich liegt zwischen 0 und 65.535.	 (16 Bitzustände)
DWORD	Doppelwort	Dieser Datentyp entspricht dem Datentyp "Wort", belegt jedoch im Datenspeicher 32 aufeinanderfolgende Bits.	
BCD-4	Vierstellige BCD-Zahl	Vierstellige BCD-Zahlen belegen 16 Bits im Datenspeicher. Jede BCD-Stelle belegt vier Bits und kann eine Zahl zwischen 0 und 9 darstellen. Die BCD-Codierung der 16 Bits ergibt einen Zahlenbereich von 0 bis 9999.	 (4 BCD-Stellen)
BCD-8	Achtstellige BCD-Zahl	Diese Zahlen belegen im Datenspeicher zwei aufeinanderfolgende 16 Bit-Adressen. Jede BCD-Stelle belegt vier Bits und stellt eine Zahl zwischen 0 und 9 dar. Die BCD-Codierung der 8-stelligen BCD-Zahl ergibt einen Zahlenbereich von 0 bis 99999999.	 (8 BCD-Stellen)

S = Vorzeichenbit (0 = positiv, 1 = negativ)

Tabelle 2-3. Datentypen (Fortsetzung)

Typ	Bezeichnung	Beschreibung	Datenformat
REAL	Gleitpunkt	Reelle Zahlen belegen 32 aufeinanderfolgende Bits (tatsächlich zwei aufeinanderfolgende 16-Bit Speicherplätze). Der Bereich der Zahlen, die in diesem Format abgelegt werden können, geht von ±1,401298E-45 bis ±3,402823E+38.	Siehe hierzu Seite 2-16.
MIXED	Gemischt	Der gemischte Datentyp kann nur bei den Multiplikations- und Divisionsfunktionen verwendet werden. Die MUL-Funktion besitzt zwei ganzzahlige Eingänge und erzeugt ein doppeltes ganzzahliges Ergebnis. Die DIV-Funktion, die ein ganzzahliges Ergebnis liefert, hat einen doppelt ganzzahligen Dividenden und einen ganzzahligen Divisor.	 

Gleitpunktzahlen

In Logicmaster 90-70 können Zahlen mit reellen Werten editiert, angezeigt, gespeichert und aufgerufen werden. Die Gleitpunktzahlen, die auch von einigen Funktionen verwendet werden, werden in dezimaler Schreibweise mit sechs signifikanten Stellen dargestellt.

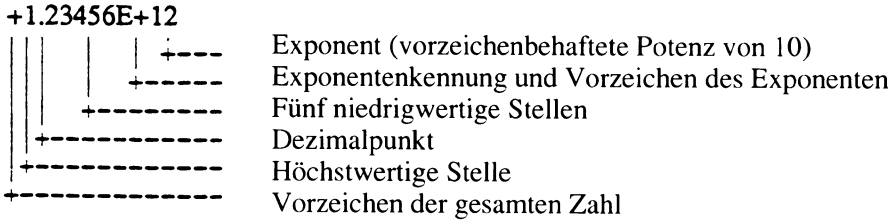
HINWEIS

In diesem Handbuch werden die beiden Begriffe "Gleitpunkt" und "reell" zur Beschreibung der in der Logicmaster Software vorhandenen Anzeigen- und Eingabefunktionen für Gleitpunktzahlen verwendet.

Bei der Logicmaster 90-70 Software werden die folgenden Formate verwendet. Zahlen im Bereich zwischen 9999999999 und 0,0001 werden mit sechs oder sieben Wertziffern ohne Exponent angezeigt. Zum Beispiel:

Eingabe	Anzeige	Beschreibung
.000123456789	+.0001234567	Zehn Stellen, davon sechs oder sieben signifikant
-12.345e-2	-.1234500	Sieben Stellen, sechs oder sieben signifikant
1234	+1234.000	Sieben Stellen, sechs oder sieben signifikant

Außerhalb des genannten Bereiches werden nur sechs Wertziffern angezeigt. Die Anzeige hat dann die Form:



Eingabe und Anzeige von Gleitpunktzahlen

Obwohl in der Mantisse sechs oder sieben für die Genauigkeit signifikante Stellen eingegeben und gespeichert werden können, zeigt die Logicmaster 90-70 Software nur die ersten sechs dieser Stelllen an. Vor der Mantisse kann ein positives oder negatives Vorzeichen stehen. Wurde kein Vorzeichen eingegeben, dann wird angenommen, daß die Gleitpunktzahl positiv ist.

Wird ein Exponent eingegeben, dann muß diesem der Buchstabe **E** oder **e** vorangestellt werden und die Mantisse muß einen Dezimalpunkt enthalten, damit die Eingabe nicht als Hexadezimalzahl mißinterpretiert wird. Vor dem Exponenten kann ein Vorzeichen eingegeben werden, ohne Eingabe wird ein positives Vorzeichen angenommen. Wird kein Exponent eingegeben, dann wird dieser mit Null angenommen. Eine Gleitpunktzahl darf keine Leerzeichen enthalten.

Zur einfacheren Verwendung werden in Befehlszeile und Eingabefeld mehrere Formate akzeptiert. Hierzu gehören ganze und dezimale Zahlen sowie Dezimalzahlen mit nachfolgendem Exponenten. Diese Zahlen werden zur Anzeige in eine Standardform umgewandelt, wenn Sie nach der Eingabe die ENTER-Taste drücken.

Die nachstehende Tabelle zeigt Beispiele zulässiger Eingaben von Gleitpunktzahlen sowie deren Anzeige.

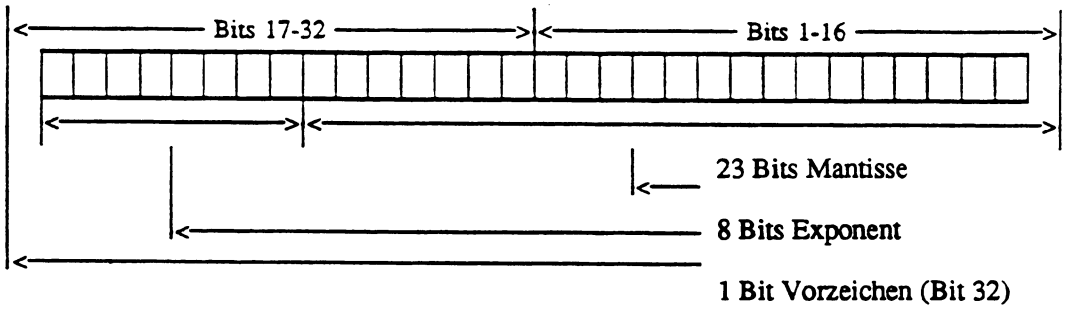
Eingabe	Anzeige
250	+250.0000
+4	+4.000000
-2383019	-2383019.
34.	+34.00000
-.0036209	-.003620900
12.E+9	+1.20000E+10
-.0004E-11	-4.00000E-15
731.0388	+731.0388
99.20003E-29	+9.92000E-28

Die nachstehende Tabelle zeigt Beispiele unzulässiger Eingaben von Gleitpunktzahlen.

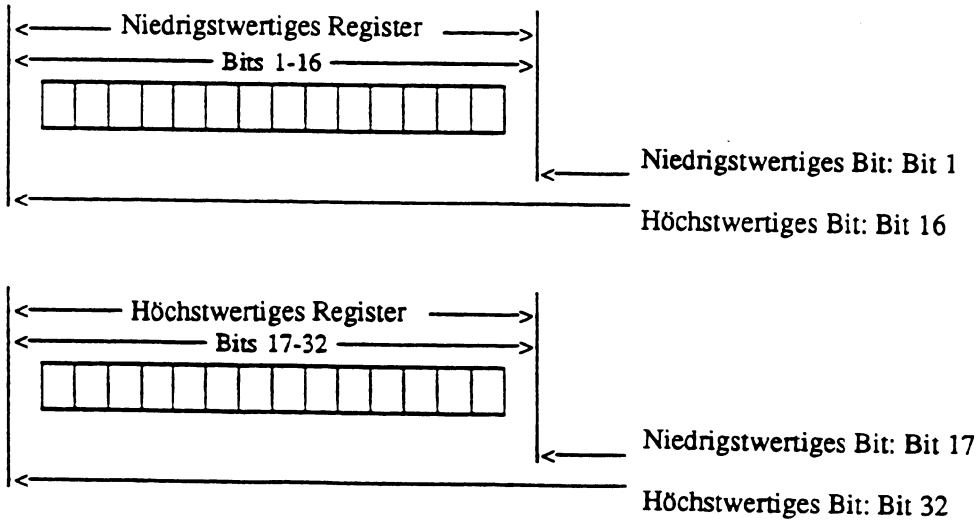
Unzulässige Eingabe	Erläuterung
-433E23	Dezimalpunkt fehlt.
10e-19	Dezimalpunkt fehlt.
10. e19	Die Mantisse darf keine Leerzeichen enthalten. Dieser Wert wird als 10.e0 übernommen und es erscheint eine Fehlermeldung.
4.1e 19	Der Exponent darf keine Leerzeichen enthalten. Dieser Wert wird als 4.1e0 übernommen und es erscheint eine Fehlermeldung.

Internes Format der Gleitpunktzahlen

Die Gleitpunktzahlen werden im einfachgenauen IEEE-Standardformat abgelegt. Dieses Format belegt 32 Bits, die in zwei (benachbarte) 16-Bit SPS-Register umgesetzt werden. Die Codierung der Bits ist in den nachstehenden Diagrammen dargestellt.



Die nachstehende Abbildung zeigt die Registerbelegung durch eine einzelne Gleitpunktzahl. Belegt in dieser Abbildung die Gleitpunktzahl zum Beispiel die Register R5 und R6, dann ist R5 das niedrigwertige und R6 das höchstwertige Register.



Werte der Gleitpunktzahlen

Verwenden Sie die nachstehende Tabelle, wenn Sie den Wert einer Gleitpunktzahl aus einer in zwei Registern gespeicherten Binärzahl berechnen wollen.

Exponent (e)	Mantisse (f)	Wert der Gleipunktzahl
255	Ungleich Null	Keine zulässige Zahl
255	0	$-1^s * \infty$
0 < e < 255	Beliebiger Wert	$-1^s * 2^{e-127} * 1.f$
0	Ungleich Null	$-1^s * 2^{e-126} * 1.f$
0	0	0

f = Mantisse
e = Exponent
s = Vorzeichenbit
* = Multiplikationsoperator

Betrachten wir zum Beispiel die Gleitpunktzahl 12.5. Die binäre Gleitpunktdarstellung (IEEE) dieser Zahl ist:

01000001 01001000 00000000 00000000

oder, im Hexadezimalformat, 41480000. Das höchstwertige Bit (das Vorzeichenbit) ist Null (s=0). Die nächsten acht höchstwertigen Bits sind 10000010 oder, in Dezimalform, 130 (e=130).

Die Mantisse wird als dezimale Binärzahl mit dem Dezimalpunkt vor dem höchstwertigen der 23 Bits gespeichert. Hierdurch ist das höchstwertige Bit in der Mantisse ein Vielfaches von 2^{-1}, das nächste Bit in der Wertigkeit ist ein Vielfaches von 2^{-2}, usw. bis zum niedrigstwertigen Bit, das ein Vielfaches von 2^{-23} darstellt. Die letzten 23 Bits (der Mantisse) sind 100100 00000000 00000000. Der Wert der Mantisse beträgt dann .5625 (d.h. $2^{-1} + 2^{-4}$).

Da e größer 0 und kleiner 255 ist, verwenden wir die dritte Formel aus der vorstehenden Tabelle:

Zahl = -1<sup typ=s> * 2<sup typ=e-127> * 1.f
= -1<sup typ=0 * 2<sup typ=130-127> * 1.5625
= 1 * 2<sup typ=3> * 1.5625
= 8 * 1.5625
= 12.5

Wir sehen daraus, daß die vorstehende Binärdarstellung richtig ist.

Der Wertebereich der Zahlen, die in diesem Format gespeichert werden können, liegt zwischen ±1.401298E-45 und ±3.402823E+38.

System-Statusreferenzen

Die System-Statusreferenzen der SPS Serie 90 sind den Speicherbereichen %S, %SA, %SB und %SC zugeordnet und besitzen jeweils eine symbolische Adresse. Beispiele für Zeitreferenzen sind T_10MS, T_100MS, T_SEC und T_MIN. Beispiele für Funktionsreferenzen sind fst_scn, alw_on und alw_off.

HINWEIS

%S-Bits können nur gelesen werden. %SA-, %SB- oder %SC-Bits können gelesen und verändert werden.

In der nachstehenden Tabelle sind alle in einem Anwenderprogramm möglichen Systemreferenzen zusammengefaßt. Bei der Eingabe eines Programms können entweder die Referenz oder die symbolische Adresse verwendet werden. Ausführlichere Fehlerbeschreibungen und weitere Informationen zur Fehlerbehebung finden Sie in GFK-0262.

Obwohl Sie diese Spezialbezeichnungen mit beliebigem Kontext verwenden können, gibt die Logicmaster Software die folgende Meldung aus, wenn Sie eine dieser Bezeichnungen für eine andere Anwendung einsetzen (z.B. als Programmblockbezeichnung, Ordnerbezeichung o.ä.):

Reuse system reserved nickname _____ ? (Y/N)

[Wollen Sie die reservierte Bezeichnung _____ nochmals verwenden ? (Ja/Nein)]

Tabelle 2-4. System-Statusreferenzen

Referenz	Symbolische Adresse	Definition
%S0001	FST_SCN	Der aktuelle Zyklus ist der erste Zyklus.
%S0002	LST_SCN	Wird bei der SPS Serie 90-70 nicht verwendet.
%S0003	T_10MS	Zeitkontakt 0,01 Sekunde.
%S0004	T_100MS	Zeitkontakt 0,1 Sekunde.
%S0005	T_SEC	Zeitkontakt 1,0 Sekunde.
%S0006	T_MIN	Zeitkontakt 1,0 Minute.
%S0007	ALW_ON	Immer EIN.
%S0008	ALW_OFF	Immer AUS.
%S0009	SY_FULL	Wird gesetzt, wenn die SPS-Fehlertabelle voll ist. Wird gelöscht, wenn ein Eintrag aus der SPS-Fehlertabelle entfernt wurde oder wenn die SPS-Fehlertabelle gelöscht wurde.
%S0010	IO_FULL	Wird gesetzt, wenn die E/A-Fehlertabelle voll ist. Wird gelöscht, wenn ein Eintrag aus der E/A-Fehlertabelle entfernt wurde oder wenn die E/A-Fehlertabelle gelöscht wurde.
%S0011	OVR_PRE	Wird gesetzt, wenn im %I-, %Q-, %M- oder %G-Speicher ein Override-Zustand besteht.
%S0012	FRC_PRE	Wird gesetzt, wenn ein Genius-Punkt gesetzt wurde.
%S0013	PRG_CHK	Wird gesetzt, wenn die Überprüfung des Hintergrundprogramms läuft.
%S0121	FST_EXE	Dieser Progammblock wurde im aktuellen Zyklus erstmalig aufgerufen.
%SA0001	PB_SUM	Wird gesetzt, wenn die für das Anwenderprogramm berechnete Prüfsumme nicht mit der Referenz-Prüfsumme übereinstimmt. War die Ursache ein vorübergehender Fehler, dann kann das Bit gelöscht werden, indem das Programm erneut in die CPU gespeichert wird. War die Ursache ein RAM-Fehler, dann muß die CPU ersetzt werden.

Tabelle 2-4. System-Statusreferenzen (Fortsetzung)

Referenz	Symbolische Adresse	Definition
%SA0002	OV_SWP	Wird gesetzt, wenn die SPS erkennt, daß der letzte Zyklus länger als die vom Anwender vorgegebene Zeit war. Wird gelöscht, wenn die SPS erkennt, daß der letzte Zyklus nicht länger als die vom Anwender angegebene Zeit war. Wird auch beim Übergang von STOP nach RUN gelöscht.
%SA0003	APL_FLT	Wird gesetzt, wenn im Anwenderprogramm ein Fehler auftritt. Wird beim Übergang von STOP nach RUN gelöscht.
%SA0009	CFG_MM	Wird gesetzt, wenn beim Einschalten oder Speichern der Konfiguration eine Diskrepanz in der Konfiguration festgestellt wird. Wird beim Einschalten der SPS gelöscht, wenn keine Diskrepanz mehr festgestellt wird oder beim Speichern einer Konfiguration, die mit der Hardware übereinstimmt.
%SA0010	HRD_CPU	Wird gesetzt, wenn die Diagnosefunktionen ein Hardwareproblem erkennen. Wird gelöscht, wenn das CPU-Modul ausgewechselt wird.
%SA0011	LOW_BAT	Wird gesetzt, wenn die Batteriespannung abfällt. Wird gelöscht, wenn die Batterie ausgewechselt und die CPU mit normaler Batteriespannung wieder eingeschaltet wird.
%SA0012	LOS_RCK	Wird gesetzt, wenn der Datenverkehr zwischen einem Erweiterungschassis und der CPU unterbrochen wird. Wird gelöscht, wenn das Problem behoben und die Versorgungsspannung des Chassis aus- und wieder eingeschaltet wurde.
%DA0013	LOS_IOC	Wird gesetzt, wenn der Datenverkehr zwischen einem Genius-Buscontroller und der CPU unterbrochen wird. Wird gelöscht, wenn das Problem behoben und die Versorgungsspannung des Chassis, in dem das Modul enthalten ist, aus- und wieder eingeschaltet wurde.
%SA0014	LOS_IOM	Wird gesetzt, wenn die Kommunikation zwischen einem E/A-Modul und der SPS CPU ausfällt. Wird gelöscht, indem die Versorgungsspannung zu dem betreffenden Chassis abgeschaltet und das Modul ersetzt wird.
%SA0015	LOS_SIO	Wird gesetzt, wenn die Kommunikation zwischen einem Zusatzmodul und der CPU ausfällt. Wird gelöscht, wenn die Versorgungsspannung zu dem betreffenden Chassis abgeschaltet und das Modul ersetzt wird.
%SA0017	ADD_RCK	Wird gesetzt, wenn dem System ein Erweiterungschassis hinzugefügt wurde. Wird gelöscht, wenn die Versorgungsspannung des Chassis aus- und wieder eingeschaltet wird und die Konfiguration nach dem Speichern mit der Hardware übereinstimmt.
%SA0018	ADD_IOC	Wird gesetzt, wenn dem System ein Genius-Buscontroller hinzugefügt wurde. Wird gelöscht, wenn die Versorgungsspannung des Chassis, das das Modul enthält, aus- und wieder eingeschaltet wird und die Konfiguration nach dem Speichern mit der Hardware übereinstimmt.
%SA0019	ADD_IOM	Wird gesetzt, wenn ein E/A-Modul in einem Chassis hinzugefügt wird. Wird gelöscht, wenn die Konfigurationsdatei nach dem Speichern mit der Hardware übereinstimmt und die Versorgungsspannung zu dem betreffenden Chassis aus- und eingeschaltet wird.
%SA0020	ADD_SIO	Wird gesetzt, wenn ein Zusatzmodul in einem Chassis hinzugefügt wird. Wird gelöscht, wenn die Konfigurationsdatei nach dem Speichern mit der Hardware übereinstimmt und die Versorgungsspannung zu dem betreffenden Chassis aus- und eingeschaltet wird.
%SA0022	IOC_FLT	Wird gesetzt, wenn ein Genius-Buscontroller einen Busfehler, einen globalen Speicherfehler oder einen Hardwarefehler beim E/A-Controller meldet. Wird gelöscht, wenn die Konfigurationsdatei nach dem Speichern mit der Hardware übereinstimmt und die Versorgungsspannung zu dem betreffenden Chassis aus- und eingeschaltet wird.
%SA0023	IOM_FLT	Wird gesetzt, wenn ein E/A-Modul einen Schaltkreis- oder Modulfehler meldet. Wird gelöscht, wenn die Konfigurationsdatei nach dem Speichern mit der Hardware übereinstimmt und die Versorgungsspannung zu dem betreffenden Chassis aus- und eingeschaltet wird.
%SA0027	HRD_SIO	Wird gesetzt, wenn in einem Zusatzmodul ein Hardwarefehler erkannt wurde. Wird gelöscht, indem das Modul ausgewechselt und die Versorgungsspannung des betreffenden Chassis aus- und eingeschaltet wird.
%SA0029	SFT_IOC	Softwarefehler im E/A-Controller.

Tabelle 2-4. System-Statusreferenzen

Referenz	Symbolische Adresse	Definition
%SA0031	SFT_SIO	Wird gesetzt, wenn ein Genius-Buscontroller einen internen Softwarefehler erkennt. Wird gelöscht, indem die Konfigurationsdatei aktualisiert und die Versorgungsspannung des betreffenden Chassis aus- und eingeschaltet wird.
%SA0032	SBUS_ER	Wird gesetzt, wenn auf der VMEbus-Rückwandplatine ein Busfehler auftritt. Wird gelöscht, indem die Versorgungsspnanung des Hauptchassis aus- und eingeschaltet wird.
%SB0001	WIND_ER	Wird gesetzt, wenn bei konstanter Zyklusdauer für das Programmiergeräte-Window nicht genügend Zeit bleibt. Wird gelöscht, wenn die SPS erkennt, daß im vorhergegangenen Zyklus genügend Zeit übrig war, damit das Programmiergeräte-Window bei konstanter Zyklusdauer bearbeitet werden konnte.
%SB0009	NO_PROG	Wird gesetzt, wenn die CPU der SPS mit erhaltenem Speicher, aber ohne Anwenderprogramm eingeschaltet wird. Wird gelöscht, wenn die CPU mit einem geladenen Anwenderprogramm eingeschaltet wird.
%SB0010	BAD_RAM	Wird gesetzt, wenn die CPU beim Einschalten einen RAM-Fehler erkennt. Wird gelöscht, wenn die CPU beim Einschalten erkennt, daß der RAM fehlerfrei ist.
%SB0011	BAD_PWD	Wird gesetzt, wenn ein durch Paßwort geschützter Zugriff verletzt wird. Wird gelöscht, wenn die SPS-Fehlertabelle gelöscht wird.
%SB0012	NUL_CFG	Wird gesetzt, wenn versucht wurde, die SPS ohne Konfigurationsdaten in RUN-Modus zu versetzen. Wird gelöscht, wenn Konfigurationsdaten vorhanden sind, ehe die SPS in RUN-Modus versetzt wird.
%SB0013	SFT_CPU	Wird gesetzt, wenn die CPU einen nicht behebbaren Fehler in der Software erkennt. Wird gelöscht, indem die Versorgungsspannung der CPU aus- und wieder eingeschaltet wird.
%SB0014	STOR_ER	Wird gesetzt, wenn während des Abspeicherns ins Programmiergerät ein Fehler auftritt. Wird gelöscht, wenn ein Speichervorgang fehlerfrei beendet wurde.
%SB0015	GADR_MM	Wird bei SPS Serie 90-70 nicht verwendet.
%SB0016	MAX_IOC	Wird gesetzt, wenn für das System mehr als 32 E/A-Controller konfiguriert wurden. Wird gelöscht, wenn die Konfiguration geändert und in der CPU abgespeichert wird.
%SB0017	SBUS_FL	Wird gesetzt, wenn die SPS keinen Zugriff zum Bus erhält. Wird gelöscht, indem die Versorgungsspannung des Hauptchassis aus- und wieder eingeschaltet wird.
%SC0009	ANY_FLT	Wird gesetzt, wenn ein beliebiger Fehler auftritt. Wird gelöscht, wenn beide Fehlertabellen gelöscht werden.
%SC0010	SY_FLT	Wird gesetzt, wenn ein Fehler auftritt, der einen Eintrag in die SPS-Fehlertabelle verursacht. Wird gelöscht, wenn die SPS-Fehlertabelle keinen Eintrag enthält.
%SC0011	IO_FLT	Wird gesetzt, wenn ein Fehler auftritt, der einen Eintrag in die E/A-Fehlertabelle verursacht. Wird gelöscht, wenn die SPS-Fehlertabelle keinen Eintrag enthält.
%SC0012	SY_PRES	Wird gesetzt, wenn die SPS-Fehlertabelle mindestens einen Eintrag enthält. Wird gelöscht, wenn die SPS-Fehlertabelle keinen Eintrag enthält.
%SC0013	IO_PRES	Wird gesetzt, wenn die E/A-Fehlertabelle mindestens einen Eintrag enthält. Wird gelöscht, wenn die E/A-Fehlertabelle keinen Eintrag enthält.
%SC0014	HRD_FLT	Es ist ein Hardwarefehler aufgetreten. Wird gelöscht, wenn beide Fehlertabellen keinen Eintrag enthalten.
%SC0015	SFT_FLT	Es ist ein Softwarefehler aufgetreten. Wird gelöscht, wenn beide Fehlertabellen keinen Eintrag enthalten.

Fehlerreferenzen

Mit den nachstehend aufgelisteten Fehler-Summenreferenzen kann festgestellt werden, welcher spezifische Fehlertyp aufgetreten ist.

Fehlerreferenz	Beschreibung
ANY_FLT	Beliebiger Fehler im System
SY_FLT	Beliebiger Systemfehler in der SPS Serie 90
IO_FLT	Beliebiger E/A-Fehler
HRD_FLT	Beliebiger Hardwarefehler
SFT_FLT	Beliebiger Softwarefehler
SY_PRES	Zeigt einen neuen Eintrag in die SPS-Fehlertabelle an.
IO_PRES	Zeigt einen neuen Eintrag in die E/A-Fehlertabelle an.

Beim Einschalten werden die Fehler-Summenreferenzen gelöscht. Tritt ein Fehler auf, dann wird die betroffenen Referenz beim nächsten auf den Fehler folgenden Zyklus gesetzt und bleibt solange gesetzt, bis er vom Programm gelöscht wird oder bis die SPS gelöscht wird. Die Fehlerreferenz ANY_FLT wird gesetzt, wenn ein sonstiger Fehler aufgetreten ist. Die Fehlerreferenzen SY_PRES und IO_PRES werden gesetzt, wennn die Fehlertabellen (SPS bzw. E/A) einen Eintrag enthalten.

Zusätzlich zu einer Fehler-Summenreferenz werden weitere Fehlerreferenzen gesetzt, die in der nachstehenden Tabelle zusammengefaßt sind. Bei den mit einem Sternchen markierten Referenzen handelt es sich um konfigurierbare System-Fehlerreferenzen.

	SPS-Systemfehler (SY_FLT)	E/A-Fehler (IO_FLT)
Hardwarefehler (HRD_FLT	SBUS_ER Systembusfehler HRD_CPU CPU-Hardwarefehler in SPS HRD_NIO Modul-Hardwarefehler SBUS_FL Systembusfehler *	
Softwarefehler (SFT_FLT)	SFT_NIO Softwarefehler in Nicht-E/A-Modul SFT_CPU SPS-Softwarefehler * MAX_IOC Zu viele Buscontroller * STOR_ER Laden vom Programmiergerät fehlerhaft *	SFT_IOC
Sonstige Fehler	PB_SUM Programmblock-Prüfsummenfehler LOW_BAT Batteriespannung zu niedrig OV_SWP Konstante Zykluszeit überschritten SY_FULL SPS-Fehlertabelle voll I/O_FULL E/A-Fehlertabelle voll APL_FLT Fehler im Anwenderprogramm NO_PROG Kein Anwenderprogramm beim einschalten * BAD_RAM Fehlerhafter Programmspeicher * WIND_ER Unvollständige Windowbearbeitung * BAD_PWR Fehler bei Paßwortzugriff * NUL_CFG Keine Konfiguration vorhanden *	LOS_IOC Buscontroller verloren LOS_IOM E/A-Modul verloren ADD_IOC Buscontroller hinzugefügt ADD_IOM E/A-Modul hinzugefügt IOC_FLT Fehler bei Bus oder Buscontroller IOM_FLT Fehler in E/A-Modul

* Konfigurierbare System-Fehlerreferenzen

Konfigurierbare Systemfehler-Referenzen

Bei einigen Fehlern muß der SPS-Betrieb angehalten werden. Die angemessene Reaktion auf einen Fehler hängt von der Art der Anwendung ab. Sämtliche Systemfehler können einer der folgenden drei Aktionen zugeordnet werden:

Aktion	Beschreibung
Fatal	Diese Fehler stoppen das System, setzen Diagnosevariablen und werden in der Fehlertabelle eingetragen.
Diagnose	Diese Fehler bewirken keinen Systemstopp. Sie setzen Diagnosevariablen und werden in der Fehlertabelle eingetragen.
Informativ	Diese Fehler werden in der Fehlertabelle eingetragen, bewirken jedoch keine weiteren Aktionen.

Bei "nicht konfigurierbaren" Fehlern kann die durch den Fehler hervorgerufene Aktion nicht verändert werden. Bei "konfigurierbaren" Fehlern kann der Fehlertyp verändert werden, wenn dies für die Anwendung geeignet erscheint.

In den nachstehenden Tabellen werden konfigurierbare Fehler beschrieben. Nur der ursprüngliche Fehler wird protokolliert, nicht eventuell auftretende Folgefehler.

Tabelle 2-5. Konfigurierbare Systemfehler-Referenzen

Fehler	Beschreibung	Kann auch gesetzt sein
SBUS_ER (Diagnose)	Systembusfehler (das Signal BSERR* wurde auf dem VME-Bus erzeugt)	HRD_FLT SY_PRES, SY_FLT Je nach Zugriff beim Auftreten von BSERR* können noch weitere Referenzen gesetzt werden
HRD_CPU (fatal)	SPS-CPU-Hardwarefehler (z.B. Speicherfehler oder Defekt im seriellen Port).	SY_FLT, SY_PRES HRD_FLT
HRD_NIO (Diagnose)	Nicht-fataler Hardwarefehler in einem Modul des Systems (z.B. im seriellen Port eines PCM).	SY_FLT, SY_PRES HRD_FLT
SFT_IOC (Diagnose)	Nicht behebbarer Softwarefehler in einem Genius-E/A-Buscontroller.	IO_FLT, IO_PRES SFT_FLT
SFT_NIO (Diagnose)	Nicht behebbarer Softwarefehler in einem PCM oder einem LAN-Schnittstellenmodul.	SY_FLT, SY_PRES SFT_FLT
PB_SUM (fatal) LOW_BAT (Diagnose)	Programmblock-Prüfsummenfehler beim Einschalten oder im RUN-Modus. Meldung "Batteriespannung abgefallen" von CPU oder einem anderen Modul im System.	SY_FLT, SY_PRES SY_FLT, SY_PRES
OV_SWP (Diagnose)	Konstante Zykluszeit überschritten.	SY_FLT, SY_PRES

Tabelle 2-5. Konfigurierbare Systemfehler-Referenzen (Fortsetzung)

Fehler	Beschreibung	Kann auch gesetzt sein
SY_FULL IO_FULL (Diagnose)	SPS-Fehlertabelle ist voll (16 Einträge), E/A-Fehlertabelle ist voll (32 Einträge).	SY_FLT, SY_PRES IO_FLT, IO_PRES
APL_FLT (Diagnose)	Anwenderfehler	SY_FLT, SY_PRES
LOS_RCK (Diagnose)	Chassisverlust (BRM-Fehler, Spannungsausfall) oder Verlust eines konfigurierten Chassis.	SY_FLT, SY_PRES IO_FLT, IO_PRES
LOS_IOC (Diagnose)	Verlust von Buscontroller-Kanal oder Verlust eines konfigurierten Buscontrollers.	IO_FLT, IO_PRES
LOS_IOM (Diagnose)	Verlust eines E/A-Moduls (antwortet nicht) oder Verlust eines konfigurierten E/A-Moduls.	IO_FLT, IO_PRES
LOS_SIO (Diagnose)	Verlust eines anderen Modultyps (antwortet nicht) oder Verlust eines konfigurierten Moduls.	SY_FLT, SY_PRES
ADD_RCK (Diagnose)	Neues Chassis hinzugefügt oder zuvor fehlerhaftes Chassis zurückgekehrt.	SY_FLT, SY_PRES
ADD_IOC (Diagnose)	Fehlerhafter Buscontroller ist nicht mehr fehlerhaft.	IO_FLT, IO_PRES
ADD_IOM (Diagnose)	Fehlerhaftes E/A-Modul ist nicht mehr fehlerhaft.	IO_FLT, IO_PRES
ADD_SIO (Diagnose)	Ein neues Nicht-E/A-Modul wurde hinzugefügt oder ein fehlerhaftes Modul ist nicht mehr fehlerhaft.	SY_FLT, SY_PRES
IOC_FLT (Diagnose)	Nicht fataler Fehler in Bus oder Buscontroller, mehr als 10 Busfehler in 10 Sekunden.	IO_FLT, IO_PRES
IOM_FLT (Diagnose)	Punkt oder Kanal auf einem E/A-Modul; partieller Ausfall des Moduls.	IO_FLT, IO_PRES
CFG_MM (fatal)	Falscher Modultyp beim Einschalten oder im RUN- Modus erkannt. Die SPS überprüft nicht die für die einzelnen Module (z.B. Genius-E/A-Blöcke) einge- stellten Parameter.	SY_FLT, SY_PRES IO_FLT, IO_PRES für E/A-Modul

Nicht konfigurierbare Systemfehler-Referenzen

Bei nicht konfigurierbaren Fehlern kann die Fehlerreaktion nicht verändert werden. Diese Fehler sind immer fatal oder informativ.

In der nächsten Tabelle werden nicht konfigurierbare Fehler beschrieben. Nur der ursprüngliche Fehler wird protokolliert, nicht eventuell auftretende Folgefehler.

Tabelle 2-6. Nicht konfigurierbare Systemfehler-Referenzen

Fehler	Beschreibung	Ergebnis
SBUS_FL (fatal)	Systembus-Fehler; die CPU konnte nicht auf den VME-Bus zugreifen. Fehler BUSGRT*NMI.	Dieser Fehler setzt keine Referenzen, die CPU wird angehalten.
NO_PROG (informativ)	Beim Einschalten ist kein Anwenderprogramm vorhanden. Darf nur beim ersten Einschalten der SPS auftreten, oder beim Ausfall des batteriegepufferten RAMs, der das Programm enthält.	Setzt keine Referenzen. Die SPS geht nicht in RUN-Modus, STOP-Modus-Zyklus wird weitergefahren, bis ein zulässiges Programm geladen wird (kann ein Null-Programm sein, das nichts ausführt).
BAD_RAM (fatal)	Fehlerhafter Programmspeicher beim Einschalten. Programm konnte nicht gelesen werden bzw. Prüfsummentest war fehlerhaft.	Setzt keine Referenzen.
WIND_ER (informativ)	Window-Abschlußfehler. Bedienung des Kommunikationswindows wurde vorzeitig abgebrochen, um nächsten Zyklus zu starten. Tritt bei konstantem Zyklus auf.	Setzt auch die folgenden Referenzen: SY_FLT, SY_PRES.
BAD_PWD (informativ)	Wechsel der Privilegebenen-Anforderung zu einer (Informations-) Schutzebene wurde abgelehnt: falsches Paßwort.	Setzt: SY_FLT, SY_PRES.
NUL_CFG (fatal)	Beim Ubergang in RUN-Modus ist keine Konfiguration vorhanden.	Setzt keine Referenzen. Lauf ohne Konfiguration entspricht Aufheben der E/A-Aktualisierung.
SFT_CPU (fatal)	CPU-Softwarefehler. Ein nicht behebbarer Fehler wurde in der CPU erkannt. Ursache kann abgelaufene Zeitüberwachung (Watchdog) sein.	Die SPS geht unmittelbar in Fehlerzyklus-Modus. Die einzige erlaubte Aktivität ist die Kommunikation mit dem Programmiergerät. Zum Löschen muß die SPS-Spannung aus- und eingeschaltet werden. Setzt auch die Referenzen SY_FLT, SY_PRES, SFT_FLT.
MAX_IOC (fatal)	Die maximale Anzahl Buscontroller wurde überschritten. Die SPS Serie 90 unterstützt 32 Buscontroller.	Setzt auch die Referenzen: SY_FLT, SY_PRES SFT_FLT
STOR_ER (fatal)	Laden der Daten vom Programmiergerät zur SPS fehlerhaft. Daten in der SPS können verstümmelt sein.	SPS geht nicht in RUN-Modus. Dieser Fehler wird beim Einschalten nicht gelöscht. Eingriff ist erforderlich zur Behebung. Setzt auch die Referenzen SY_FLT, SY_PRES.

Fehlersuchreferenzen (Chassis, Steckplatz, Bus)

Die Logicmaster 90-70 Software stellt reservierte Fehlerbezeichnungen für die einzelnen Chassis, Steckplätze, Busse und Module zur Verfügung, die bei konfigurierter Hardware mit Funktionen belegt sind. Werden diese Bezeichnungen bei den Kontaktanweisungen FAULT und NOFLT programmiert, dann kann ein Programm zur Loaklisierung der mit den konfigurierten Chassis und Modulen zusammenhängenden Fehlern ablaufen.

Diese symbolischen Fehleradressen können nur im Zusammenhang mit den Kontakten FAULT und NOFLT programmiert werden. Die reservierten Fehlernamen sind immer vorhanden. Es ist nicht erforderlich, eine spezielle Option freizugeben (z.B. Punktfehler).

Diese Fehlerbezeichnungen entsprechen weder %SA, %SB, %SC noch sonstigen Referenztypen. Über den Programmeditor oder beim Ausdruck wird nur die symbolische Adresse angezeigt, zu den Fehlerreferenzen gibt es keine Referenztabellen-Menüs.

Die symbolische Fehleradresse eines Chassis wird im Format RACK_0r dargestellt, wobei r die Chassisnummer (0 bis 7) ist. RACK_01 steht zum Beispiel für Chassis 1.

Die symbolische Fehleradresse eines Steckplatzes wird im Format SLOT_rs dargestellt, wobei r die Chassisnummer (0 bis 7) und s die Steckplatznummer (0 bis 9) ist. SLOT_25 steht z.B. für Chassis 2, Steckplatz 5.

Die symbolische Fehleradresse eines Busses wird im Format BUS_rsb dargestellt, wobei r die Chassisnummer (0 bis 7), s die Steckplatznummer (0 bis 9) und b die Busnummer (1 oder 2) ist. BUS_341 steht z.B. für Chassis 3, Steckplatz 1, Bus 1.

HINWEIS

Derzeit ist nur 1 als Busnummer zugelassen, die Hardware für Bus Nr. 2 ist noch nicht lieferbar.

Die symbolische Fehleradresse eines Moduls wird im Format M_rsbmm dargestellt, wobei r die Chassisnummer (0 bis 7), s die Steckplatznummer (0 bis 9), b die Busnummer (1 oder 2) und mm die Modulnummer (00 bis 31) ist. M_46128 steht z.B. für Chassis 4, Steckplatz 6, Bus 1, Modul 28.

Beim Einschalten werden alle Fehler in der SPS gelöscht. Wird ein Fehler eingetragen, dann schaltet die SPS den Zustand der betroffenen Referenz(en) um. Der Zustand der Fehlerreferenz bleibt solange erhalten, wie der Fehler besteht. Wird die SPS gelöscht, dann werden alle Fehler auf ihren Ausgangszustand gesetzt.

Diese Fehlerreferenzen werden nur zur Information gesetzt. Die SPS unterbricht ihren Betrieb nicht, wenn eine dieser Diagnosevariablen gesetzt wird.

Die Fehlerreferenzen sind kaskadierend. Tritt zum Beispiel bei Modul 29 im Steckplatz 6 von Chassis 5 am Bus 1 ein Fehler auf, dann werden die folgenden Fehlerreferenzen gesetzt: RACK_05, SLOT_56, BUS_561 und M_56129. Das Problem mit dem Modul wird jedoch nur durch einen einzigen Eintrag in der Fehlertabelle beschrieben, die Fehlertabelle enthält in diesem Fall keine Einträge bezüglich Chassis, Steckplatz oder Bus.

Fehlerkontakte

Mit den Kontakten (-[FAULT]-) und (-[NOFLT]-) können Fehler in diskreten oder analogen Maschinenreferenzen festgestellt werden. Diese Kontakte können zwar gesetzt, jedoch nicht überschrieben werden. Die nachstehende Tabelle zeigt die Funktion dieser beiden Kontakte.

Zustand		[NOFLT]
Fehler vorhanden	Geschlossen	Offen
Kein Fehler vorhanden	Offen	Geschlossen

Grenzwertkontakte

Mit den Kontakten für obere (-[HIALR]-) und untere (-[LOALR]-) Grenzwerte können Sie Grenzwerte im Zusammenhang mit Analogreferenzen überwachen. Zuerst muß jedoch die Verwendung von Punktfehlern in der Konfigurationssoftware freigegeben werden, indem im Konfigurations-Hauptmenü F2 [CPU-Konfiguration] und danach im CPU-Konfigurationsmenü F4 [Speicherbelegung und Punktfehler freigeben] aufgerufen wird. Schalten Sie die Einstellung für ***Point Fault Reference*** [Punktfehler-Referenz] von DISABLED [gesperrt] auf ENABLED [freigegeben] um.

Punktfehler

Obwohl Punktfehler durch den Ausfall einer zugehörigen übergeordneten internen Hardware (z.B. Ausfall eines E/A-Controllers oder Chassis) gesetzt werden, gehören sie doch zu externen E/A-Fehlern. Punktfehler können nur dann verwendet werden, wenn sie zuvor in der Konfigurationssoftware freigegeben wurden. Rufen Sie hierzu im Konfigurations-Hauptmenü F2 [CPU-Konfiguration] und danach im CPU-Konfigurationsmenü F4 [Speicherbelegung und Punktfehler freigeben] auf. Schalten Sie die Einstellung für ***Point Fault Reference*** [Punktfehler-Referenz] von DISABLED [gesperrt] auf ENABLED [freigegeben] um.

Nach der Freigabe werden eine Boolsche Referenz für jeden diskreten E/A-Punkt und eine Bytereferenz für jeden analogen E/A-Punkt zugeordnet. Der hierfür benötigte Speicherplatz geht zu Lasten des Anwenderprogramms. Jedesmal, wenn ein Punktfehler erzeugt wird, wird die E/A-Fehlerreferenz auf Systemebene (IO_FLT) gesetzt. Die vorstehend beschriebenen Kontakte (-[FAULT]-) und (-[NOFLT]-) ermöglichen den Zugriff auf die Punktfehler.

Programmkommentare

Programmkommentare sind zusätzliche erläuternde Texte in einem Programm, durch die das Programm übersichtlicher wird und einfacher zu verstehen ist. Bei der Logicmaster 90-70 Software sind folgende Programmkommentare möglich:

Tabelle 2-7. Programmkommentare

Typ	Beschreibung
Symbolische Adresse	Eine zusätzlich Kennung aus 1 bis 7 Zeichen, die einer Programmreferenz wahlweise beigegeben werden kann. Die für eine symbolische Adresse zulässigen Zeichen sind die Buchstaben A bis Z, die Zahlen von 0 bis 9, der Unterstrich und die Sonderzeichen +, -, %, #, @, <, >, = und &. Eine symbolische Adresse muß immer mit einem Buchstaben beginnen, Groß- und Kleinschreibung wird nicht beachtet. So kann zum Beispiel der Referenz %00001 die symbolische Adresse SWITCH1 oder switch1 zugeordnet werden.
Referenzbeschreibung	Ein zusätzlicher Text von max. 32 Zeichen (in Anführungszeichen), der wahlweise einer Maschinenreferenz zugeordnet werden kann. Eine Referenzbeschreibung kann mit oder ohne symbolische Adresse verwendet werden.
Kommentar	Längere Textblöcke (Strompfad-Kommentare). Ein Kommentar kann aus max. 2048 Zeichen bestehen. Der Kommentartext erscheint auf dem Bildschirm, wenn Sie den Cursor auf den zugehörigen Strompfad setzen und die Taste F10 (Zoom) drücken. Der Kommentar kann auch als Teil des Kontaktplanprogramms ausgedruckt werden.

Weitere Informationen zum Gebrauch von Programmkommentaren finden Sie in Kapitel 3, Abschnitt 2 von GFK-0263.

Abschnitt 3 Funktionsblock-Struktur

Jeder Pfad des Kontaktplans besteht aus einer oder mehreren Programmanweisungen, die entweder einfache Relais oder aber komplexere Funktionen sein können.

Format der Kontaktplanrelais

Die Logicmaster 90-70 Software beinhaltet mehrere Typen von Relaisfunktionen, die eine einfache Steuerung von Programm und Stromfluß bewirken. Beispiele hierfür sind der Schließerkontakt (-| |-) oder die negierte Spule (-|/|-). Relaiskontakte und Spulen habe jeweils einen Eingang und einen Ausgang, die zusammen den logischen "Stromfluß" durch den Kontakt bzw. die Spule bewirken.

Jedem Relaiskontakt und jeder Spule muß eine Referenz zugeteilt werden, die bei der Relaisauswahl eingegeben wird. Bei einem Kontakt gibt die Referenz den Eingang an, der den Stromfluß in den Kontakt bestimmt. Ist beim folgenden Beispiel die Eingangsreferenz %I00122 wahr, dann fließt Strom durch diesen Relaiskontakt.

```
%I00122
--| |--
```

Bei einer Spule gibt die Referenz den Ausgang an, der gesteuert wird, wenn Strom in die Spule fließt. Im nächsten Beispiel wird die Ausgangsreferenz %Q00004 durchgeschaltet, wenn Strom in die linke Seite der Spule fließt.

```
%Q00004
--( )--
```

Die Logicmaster 90 Software besitzt eine integrierte Spulenüberprüfungsfunktion, die verhindert, daß die gleiche %Q- oder %M-Referenz für mehrere Ausgänge verwendet wird. Diese Funktion kann abgeschaltet werden (siehe GFK-0263).

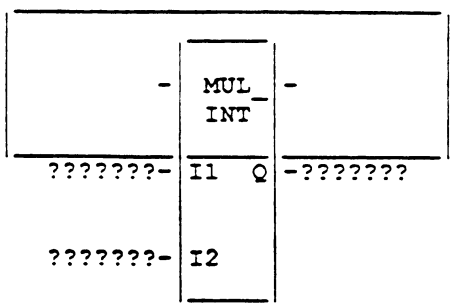
Format der Programmfunktionsblöcke

Einige Funktionen sind sehr einfach, wie z.B. die Hauptsteuerrelaisfunktion (MCR), bei der die abgekürzte Bezeichnung der Funktion in Klammern angegeben wird:

```
-[ MCR ]-
```

Andere Funktionen sind komplexer und besitzen mehrere Stellen, über die die von der Funktion benötigten Daten eingegeben werden können.

Der nachstehend abgebildete Funktionsblock ist "Multiplikation" (MUL). Seine Teile sind für zahlreiche Programmfunktionen in Logicmaster typisch. Im oberen Teil des Rechtecks ist die Funktionsbezeichnung oder ein Datentyp (hier: ganze Zahl mit Vorzeichen) angegeben.



Dies sind der "Name" der Funktion (MULtiplizieren) und der Datentyp (INT). INT steht für INTEGER (ganze Zahl) mit Vorzeichen und gibt Typ und Größe der Daten an, die verarbeitet werden sollen.

HINWEIS

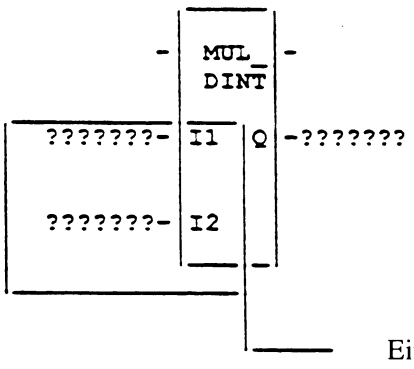
Es gibt zwei Funktionsblocktypen, von denen der eine lokale Daten verarbeitet, während der andere die Daten lediglich weiterleitet. Das Wort "Funktionsblock" wird in diesem Handbuch als übergeordneter Begriff verwendet, der sich auf beide Typen bezieht. Die in diesem Handbuch beschriebenen Funktionsblöcke nehmen Eingangsdaten auf, verändern diese aber nicht, sondern tragen lediglich das Ergebnis der Operation in den Ausgangsparameter ein.

Bei zahlreichen Programmfunktionen können Sie den Datentyp für die Funktion nach der Funktion selbst auswählen. So können Sie z.B. den Datentyp für die vorstehende Multiplikationsfunktion abändern in doppeltgenaue ganze Zahlen mit Vorzeichen. Weitere Informationen zu Datentypen finden Sie weiter vorne in diesem Kapitel.

Datenfluß zu und von einer Funktion

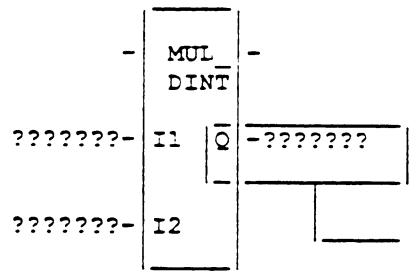
Jede Linie, die auf der linken Seite in den Fuktionsblock hineingeht, stelle einen Eingang zu dieser Funktion dar. Jede Linie, die auf der rechten Seite den Funktionsblock verläßt, steht für einen Ausgang.

Bei den Fragezeichen auf der linken Seite einer Funktion müssen Sie entweder die Daten selbst angeben oder aber eine Referenzadresse, bei der die Daten gefunden werden können.



Eingangsparameter (I1 und I2) des Funktionsblocks.

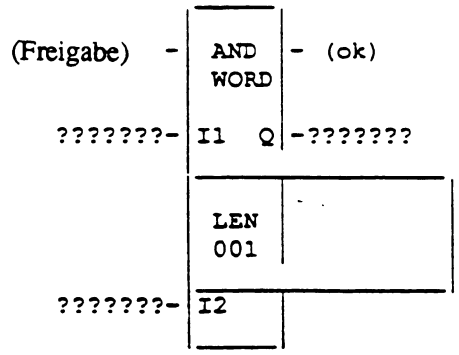
Bei den Fragezeichen auf der rechten Seite einer Funktion geben Sie normalerweise eine Referenzadresse für die von der Funktion ausgegebenen Daten an.



Ausgangsparameter (Q) für den Funktionsblock.

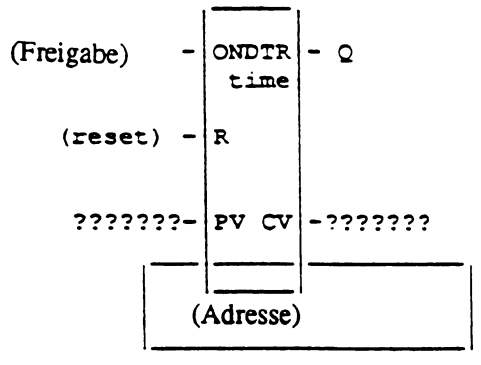
Die meisten Programmfunktionen verändern nicht die Eingangsdaten, sondern legen das Ergebnis der Operation in einer Ausgangsreferenz ab.

Bei Funktionen zur Verarbeitung von Bitfolgen kann eine Länge eingestellt werden. Beim nachstehenden Funktionsblock kann für die logische UND-Funktion eine Zeichenfolgenlänge von maximal 256 Worten oder Doppelworten eingestellt werden.



Die für den Funktionsblock eingestellte Länge.

Zähler- und Zeitgliedfunktionen benötigen eine Adreßreferenz, die die Anfangsadresse von drei Worten (Registern) enthält, in denen Istwert, Sollwert und ein Steuerwort für die Funktion abgelegt sind. Diese Referenz liegt direkt unterhalb des Funktionsblocks:

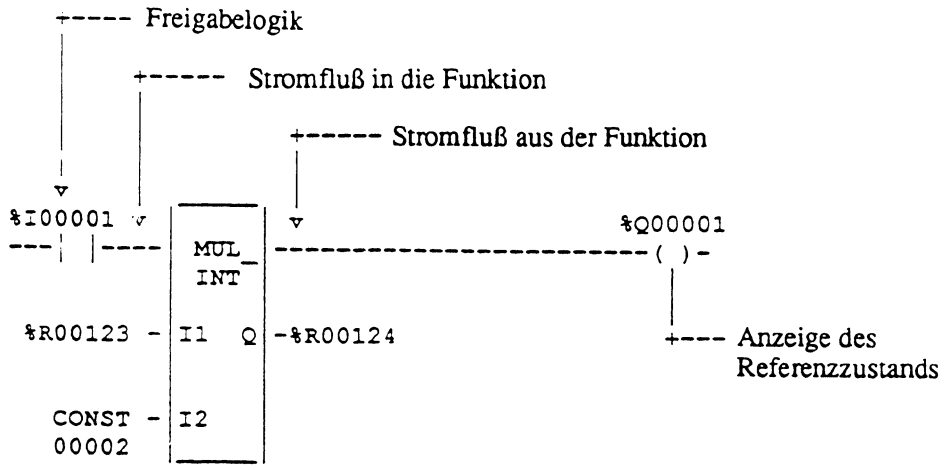


Adreßreferenz für den Funktionsblock.

Weitere Informationen über die von Zählern und Zeitgliedern benötigten Funktionsblockdaten finden Sie in Kapitel 4 dieses Handbuches.

Stromfluß zu und von einer Funktion

Der Strom fließt oben links in die Funktion. Häufig wird zur Steuerung des Stromflusses in die Funktion eine Freigabelogik verwendet; ist dies nicht der Fall, dann wird die Funktion unbedingt bei jedem CPU-Zyklus ausgeführt.

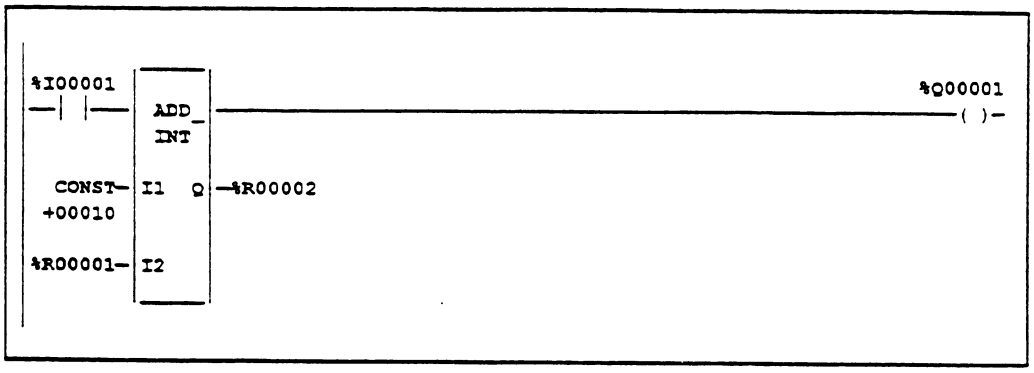


Der Strom fließt oben rechts aus der Funktion heraus und kann von dort aus wahlweise an andere Programmteile oder eine Spule weitergeführt werden. Eine Funktion schaltet den Stromfluß durch, wenn sie erfolgreich ausgeführt wurde. Bei den einzelnen Funktionsbeschreibungen in diesem Handbuch wird immer angegeben, unter welchen Bedingungen Stromfluß nach rechts stattfindet.

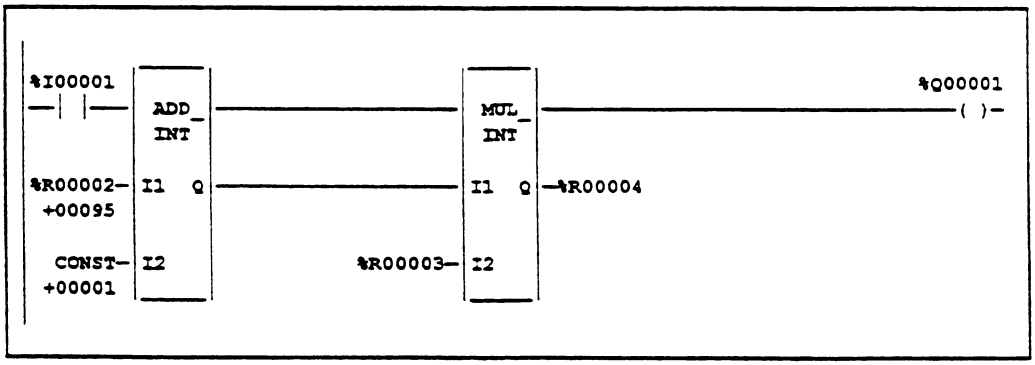
Funktionsblockparameter

In einen Funktionsblock sind drei unterschiedliche Eingaben möglich: Konstanten, Referenzen und Stromfluß.

Eine Konstante ist ein expliziter Wert. Eine Referenz ist die Adresse eines Wertes. Im folgenden Beispiel wird der Eingangswert I1 als Konstante in die Additionsfunktion eingegeben, während der Eingangswert I2 als Referenz eingegeben wird.



Stromfluß ist das Ergebnis des vorhergehenden Funktionsblocks, das unmittelbar als Eingangswert in den aktuellen Funktionsblock eingespeist wird. Ein Stromfluß ist ein unmittelbares Ergebnis, das nicht gespeichert wird, sondern direkt zum nächsten Funktionsblock oder einer Reihe von Funktionsblöcken weitergegeben wird.



Im vorstehenden Beispiel wird ein Stromfluß als Eingabe zu dem Schließerkontakt und zu den beiden Funktionen verwendet.

- Zuerst stellt der Stromfluß das Ergebnis der Einspeisung von der Stromschiene in den Schließerkontakt dar.
- Zum zweiten stellt der Stromfluß das Ergebnis des Schließerkontaktes dar, das in den Paramter EN des ADD-Funktionsblockes eigegeben wird.
- Drittens stellt der Stromfluß den OK-Ausgang des ADD-Funktionsblockes in den EN-Eingang des MUL-Funktionsblockes dar.

Es gibt nur zwei Arten von Ausgangsgrößen aus einem Funktionsblock: Referenzen und Stromfluß. Eine Konstante als Ausgangswert ist nicht möglich.

KAPITEL 3

In diesem Kapitel wird die Verwendung von Kontakten, Spulen und Verbindungen in Strompfaden eines Kontaktplanprogramms erläutert.

- Schließer- und Öffnerkontakte
- Kontakte für positive und negative Übergänge
- Kontakte zur Anzeige "Fehler" und "kein Fehler"
- Kontakte für oberen und unteren Grenzwert
- Spulen und negierte Spulen
- Remanente und negierte remanente Spulen
- Setz- und Rücksetz-Spulen
- Remanente SET- und RESET-Spulen
- Horizontal- und Vertikalverbindungen

Kontakte

Mit einem Kontakt wird der Zustand einer Maschine oder einer internen Referenz überwacht. Der Stromfluß über einen Kontakt hängt vom Zustand der überwachten Referenz und vom Kontakttyp ab.

Kontakttypen

Kontakttyp	Anzeige	Kontakt schaltet den Stromfluß nach rechts durch
Schließerkontakt	-] [-	wenn die Referenz EIN ist
Öffnerkontakt	-]/[-	wenn die Referenz AUS ist
Kontakt für positiven Übergang	-]↑[-	wenn die Referenz auf EIN umschaltet
Kontakt für negativen Übergang	-]↓[-	wenn die Referenz auf AUS umschaltet
Kontakt "Fehler"	-[FAULT]-	wenn bei der Referenz ein fehlerhafter Punkt ist
Kontakt "kein Fehler"	-[NOFLT]-	wenn bei der Referenz kein fehlerhafter Punkt ist
Kontakt "oberer Grenzwert verletzt"	-[HIALR]-	wenn die Referenz den oberen Grenzwert überschreitet
Kontakt "unterer Grenzwert verletzt"	-[LOALR]-	wenn die Ref. den unteren Grenzwert unterschreitet

Spulen

Über Spulen werden Maschinenausgänge oder interne Ausgänge gesteuert. Der Stromfluß zu einer Spule muß über eine Bedingungslogik gesteuert werden. Spulen verursachen direkt eine Aktion, sie geben keinen Stromfluß nach rechts weiter. Soll als Ergebnis eines Kontaktzustandes ein weiterer Programmteil durchlaufen werden, dann muß für diese Spule eine interne Referenz verwendet werden.

Spulen stehen immer ganz rechts in einer Programmzeile. In einem Strompfad sind bis zu acht Spulen möglich.

Die verwendeten Spulentypen hängen davon ab, welche Aktion das Programm durchführen soll. Die Zustände remanenter Spulen bleiben bei einem Ausfall der Versorgungsspannung oder beim Umschalten von STOP- in RUN-Modus erhalten.

Tabelle 3-2. Spulentypen

Spulentyp	Anzeige	Strom zu Spule	Ergebnis
Spule (Schließer)	-( )-	EIN	Schaltet Referenz EIN
		AUS	Schaltet Referenz AUS
Negiert	-(/)-	EIN	Schaltet Referenz AUS
		AUS	Schaltet Referenz EIN
Remanent	-(M)-	EIN	Schaltet Referenz EIN, remanent
		AUS	Schaltet Referenz AUS, remanent
Negiert remanent	-(/M)-	EIN	Schaltet Referenz AUS, remanent
		AUS	Schaltet Referenz EIN, remanent
Positiver Übergang	-(↑)-	AUS → EIN	Ist die Referenz AUS, geht sie für einen Zyklus auf EIN
Negativer Übergang	-(↓)-	EIN → AUS	Ist die Referenz AUS, geht sie für einen Zyklus auf EIN
SET	-(S)-	EIN	Referenz EIN, bis sie durch -(R)- auf AUS gesetzt wird.
		AUS	Keine Aktion.
RESET	-(R)-	EIN	Referenz AUS, bis sie durch -(S)- auf EIN gesetzt wird.
		AUS	Keine Aktion.
SET, remanent	-(SM)-	EIN	Referenz EIN, bis sie durch -(RM)- auf AUS gesetzt wird, remanent.
		AUS	Keine Aktion.
RESET, remanent	-(RM)-	EIN	Referenz AUS, bis sie durch -(SM)- auf EIN gesetzt wird, remanent.
		AUS	Keine Aktion.

Schließerkontakt -] [-

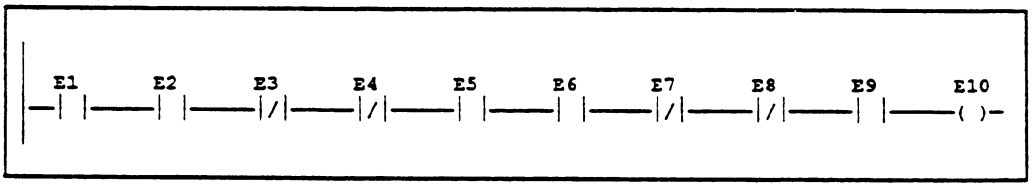
Ein Schließerkontakt arbeitet als Relais, das den Stromfluß dann weiterleitet, wenn die zugehörige Referenz EIN (1) ist.

Öffnerkontakt -]/[-

Ein Öffnerkontakt arbeitet als Relais, das den Stromfluß dann weiterleitet, wenn die zugehörige Referenz AUS (0) ist.

Beispiel:

Das folgende Beispiel zeigt einen Strompfad mit 10 Elementen, denen die symbolischen Adressen E1 bis E10 zugeordnet sind. Die Spule E10 ist dann durchgeschaltet, wenn die Referenzen E1, E2, E5, E6 und E9 EIN und die Referenzen E3, E4, E7 und E8 AUS sind.



Kontakte für positiven Übergang -|↑|-

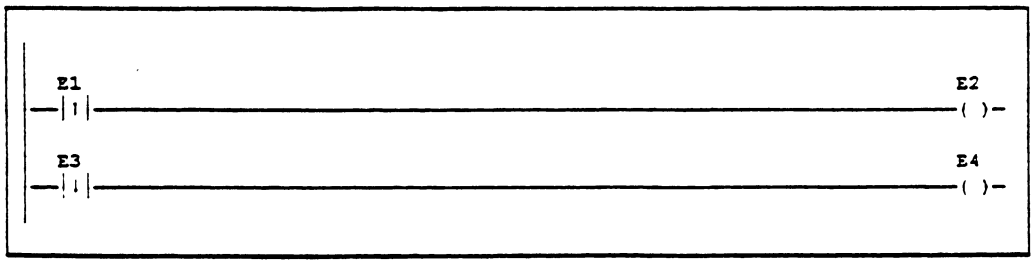
Ein Kontakt für positiven Übergang schaltet den Stromfluß weiter, wenn die zugehörige Referenz von AUS (0) nach EIN (1) umschaltet.

Kontakte für negativen Übergang -|↓|-

Ein Kontakt für negativen Übergang schaltet den Stromfluß weiter, wenn die zugehörige Referenz von EIN (1) nach AUS (0) umschaltet.

Beispiel:

Im folgenden Beispiel wird dargestellt, wie die Kontakte für positiven und negativen Übergang eingesetzt werden können. Die Spule E2 wird für einen Programmzyklus durchgeschaltet, wenn das Element E1 von AUS nach EIN umschaltet. Die Spule E4 wird für einen Programmzyklus durchgeschaltet, wenn das Element E3 von EIN nach AUS umschaltet.



Kontakt Fehler -[FAULT]-

Mit diesen Kontakten können Fehler bei diskreten oder analogen Maschinenreferenzen erkannt oder Fehler (Chassis, Steckplatz, Modul) lokalisiert werden. Die Verwendung dieser Kontakte muß bei der CPU-Konfiguration freigegeben werden, wenn die Meldung von Punktfehlern gewünscht wird. Weitere Informationen zu Punktfehlern finden Sie in Kapitel 2.

Ein Kontakt Fehler schaltet den Stromfluß nach rechts durch, wenn bei der zugehörigen Variable ein Punktfehler aufgetreten ist.

Kontakt kein Fehler -[NOFLT]-

Mit diesen Kontakten, deren Verwendung auch freigegeben werden muß, wenn die Meldung von Punktfehlern gewünscht wird, können ebenfalls Fehler bei diskreten oder analogen Maschinenreferenzen festgestellt werden. Weitere Informationen zu Punktfehlern finden Sie in Kapitel 2.

Ein Kontakt kein Fehler schaltet den Stromfluß nach rechts durch, wenn bei der zugehörigen Variable kein Punktfehler aufgetreten ist.

Kontakt oberer Grenzwert verletzt -[HIALR]-

Über diesen Kontakt kann ein mit einer analogen Referenz verknüpfter oberer Grenzwert überwacht werden. Die Verwendung der Grenzwertkontakte muß bei der CPU-Konfiguration freigegeben werden. Rufen Sie hierzu in der Konfigurationssoftware das Menü "Speicherbelegung und Punktfehlerfreigabe" auf und ändern Sie die Einstellung für die Grenzwertreferenz von DISABLED auf ENABLED.

Ein Kontakt oberer Grenzwert verletzt schaltet den Stromfluß nach rechts durch, wenn das mit der Analogreferenz verknüpfte Bit für den oberen Grenzwert "1" ist.

Kontakt unterer Grenzwert verletzt -[LOALR]-

Über diesen Kontakt kann ein mit einer analogen Referenz verknüpfter unterer Grenzwert überwacht werden. Die Verwendung der Grenzwertkontakte muß ebenfalls bei der CPU-Konfiguration freigegeben werden.

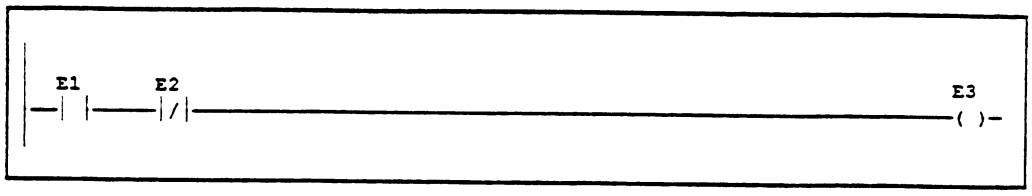
Ein Kontakt unterer Grenzwert verletzt schaltet den Stromfluß nach rechts durch, wenn das mit der Analogreferenz verknüpfte Bit für den unteren Grenzwert "1" ist.

Spule -( )-

Empfängt eine Spule Stromfluß, dann schaltet sie den zugehörigen diskreten Ausgang auf EIN. Eine Spule ist nicht remanent und kann daher nicht zusammen mit Operanden aus streng remanenten Speichern (%SA, %SB, %SC oder %G) verwendet werden.

Beispiel:

Im folgenden Beispiel wird die Spule E3 durchgeschaltet, wenn Referenz E1 EIN und Referenz E2 AUS ist.

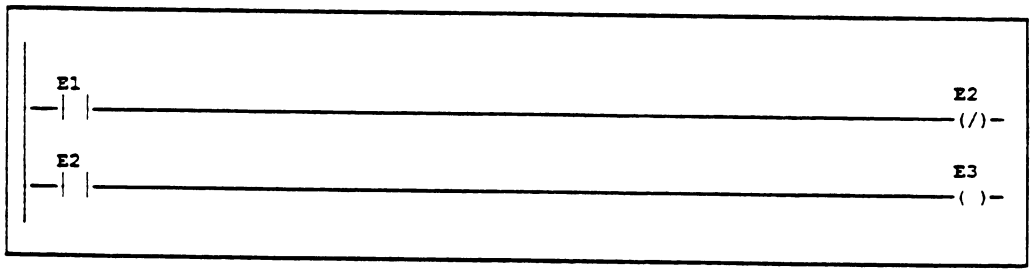


Negierte Spule -(/)-

~~Eine negierte Spule~~ schaltet einen diskreten Ausgang durch, wenn sie keinen Stromfluß empfängt. Sie ist nicht remanent und kann daher nicht zusammen mit Operanden aus streng remanenten Speichern (%SA, %SB, %SC oder %G) verwendet werden.

Beispiel:

Im folgenden Beispiel wird die negierte Spule E3 durchgeschaltet, wenn Referenz E1 AUS ist.



Remanente Spule -(M)-

Wie bei einer Spule wird bei der remanenten Spule ein diskreter Ausgang durchgeschaltet, wenn die Spule Stromfluß empfängt. Der Zustand der remanenten Spule wird über einen Stromausfall hinweg erhalten. Sie kann daher nicht mit Operanden aus streng nicht-remanenten Speicherbereichen (%T) verwendet werden.

Negierte remanente Spule -(/M)-

Die negierte remanente Spule schaltet einen diskreten Ausgang dann durch, wenn sie keinen Stromfluß empfängt. Der Zustand der remanenten Spule wird über einen Stromausfall hinweg erhalten. Sie kann daher nicht mit Operanden aus streng nicht-remanenten Speicherbereichen (%T) verwendet werden.

Spule für positive Übergänge -(↑)-

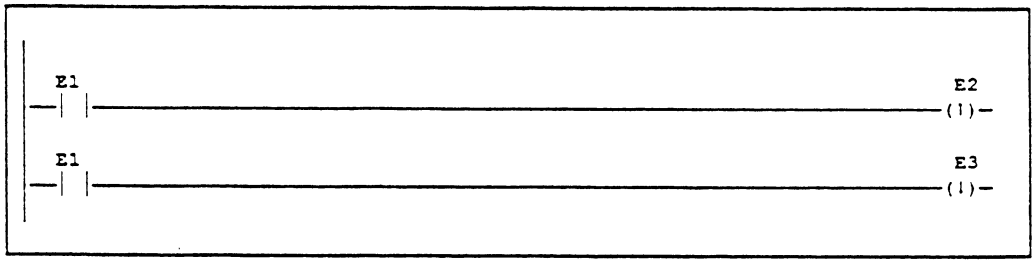
Ist der mit einer Spule für positive Übergänge verbundene Ausgang AUS, wenn die Spule Stromfluß empfängt, dann wird er für die Dauer eines Zyklus durchgeschaltet. Diese Spule kann als Wischrelais verwendet werden. Spulen für Übergänge können zusammen mit Operanden aus remanenten oder nicht remanenten Speicherbereichen (%Q, %M, %T, %G, %SA, %SB oder %SC) verwendet werden.

Spule für negative Übergänge -(↓)-

Ist der mit einer Spule für negative Übergänge verbundene Ausgang EIN, wenn der Stromfluß zur Spule unterbrochen wird, dann wird die Referenz für die Dauer eines Zyklus durchgeschaltet. Spulen für Übergänge können zusammen mit Operanden aus remanenten oder nicht remanenten Speicherbereichen (%Q, %M, %T, %G, %SA, %SB oder %SC) verwendet werden.

Beispiel:

Schaltet bei dem folgenden Beispiel die Referenz E1 von AUS nach EIN, dann fließt Strom zu den Spulen E2 und E3 und E2 wird für die Dauer eines logischen Zyklus durchgeschaltet. Schaltet E1 von EIN nach AUS, dann wird der Stromfluß zu E2 und E3 unterbrochen und die Spule E3 wird für die Dauer eines Zyklus durchgeschaltet.



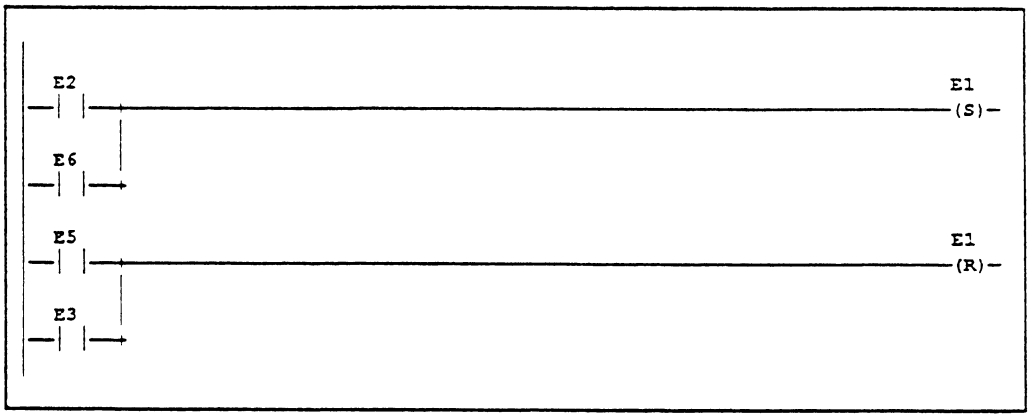
SET-Spule -(S)-

SET und RESET sind nicht-remanente Spulen, die den Zustand einer Referenz auf EIN oder AUS halten ("verriegeln") können. Wird der Stromfluß zu einer SET-Spule geschaltet, dann bleibt seine Referenz solange EIN (unabhängig davon, ob die Spule selbst Stromfluß empfängt), bis sie durch Stromfluß zu einer RESET-Spule (-(R)-) rückgesetzt wird.

RESET-Spule -(R)-

Empfängt die RESET-Spule Stromfluß, dann wird ein diskreter Maschinenausgang oder ein interner Ausgang auf AUS geschaltet und bleibt solange AUS, bis er durch eine SET-Spule wieder eingeschaltet wird. Die im Programm zuletzt bearbeitete Spule einer SET-/RESET-Kombination hat jeweils Vorrang.

Bei dem folgenden Beispiel wird die Spule E1 durchgeschaltet, wenn Referenz E2 oder E6 EIN ist. Die Spule E1 wird abgeschaltet, wenn Referenz E5 oder E3 EIN ist.



Remanente SET-Spule -(SM)-

Die Funktion remanenter SET- und RESET-Spulen ist ähnlich wie die der normalen SET- und RESET-Spulen, ihre Zustände bleiben jedoch bei einem Ausfall der Versorgungsspannung oder beim Umschalten von STOP- in RUN-Modus erhalten. Eine remanente SET-Spule schaltet einen diskreten Ausgang durch, wenn die Spule Stromfluß empfängt. Der Ausgang bleibt solange durchgeschaltet, bis er durch eine -(RM)- -Spule wieder rückgesetzt wird.

Remanente RESET-Spule -(RM)-

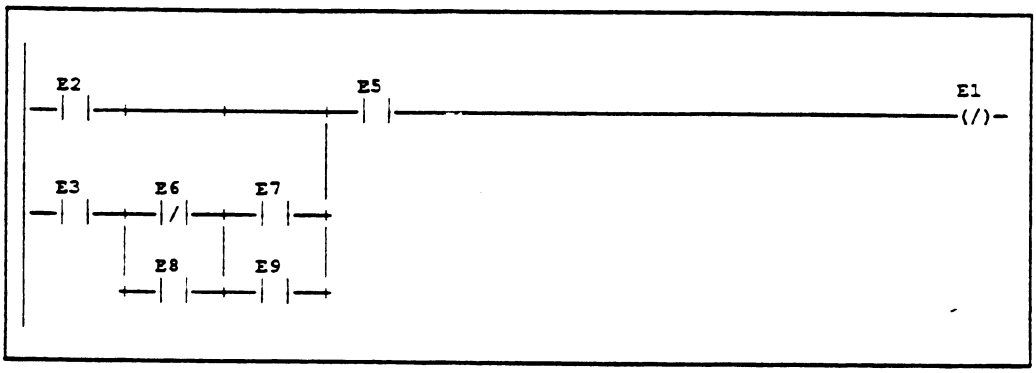
Empfängt die remanente RESET-Spule Stromfluß, dann wird ein diskreter Maschinenausgang oder ein interner Ausgang auf AUS geschaltet. Dieser Ausgang bleibt solange AUS, bis er durch eine -(SM)- -Spule wieder eingeschaltet wird. Der Zustand dieser Spule bleibt bei einem Ausfall der Versorgungsspannung oder beim Umschalten von STOP- in RUN-Modus erhalten. Siehe vorstehendes Beispiel für SET und RESET.

Verbindungen

Über Horizontal- und Vertikalverbindungen werden in einer Kontaktplanzeile Funktionen miteinander verbunden. Sie vervollständigen in einer Programmzeile den Weg für den Stromfluß von links nach rechts.

Beispiel:

Im folgenden Beispiel werden zwei Horizontalverbindungen dazu verwendet, die Kontakte E2 und E5 miteinander zu verbinden. Die Verknüpfung der Kontakte E3, E6, E7, E8 und E9 mit E2 wird über eine Vertikalverbindung hergestellt.



KAPITEL 4

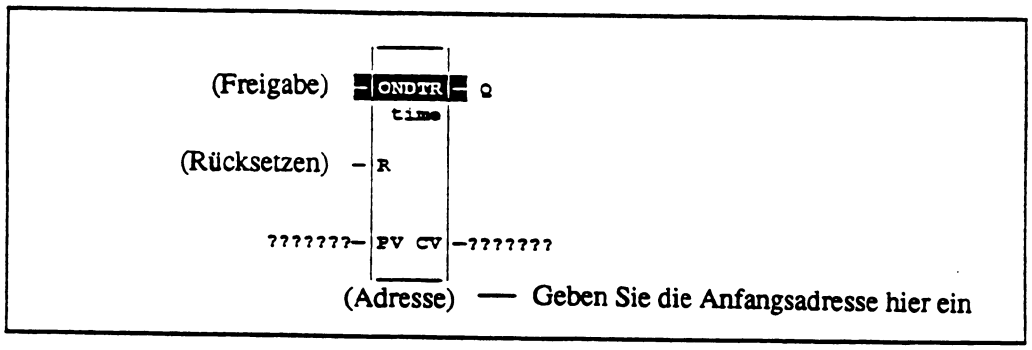
In diesem Kapitel wird die Verwendung von Auf- und Abwärtszählern sowie von Zeitgliedern zur Ein- und Ausschaltverzögerung und Zeiterfassung beschrieben. Die mit diesen Funktionen verknüpften Daten sind nullspannungssicher.

Speicherbedarf für Zeitglieder und Zähler

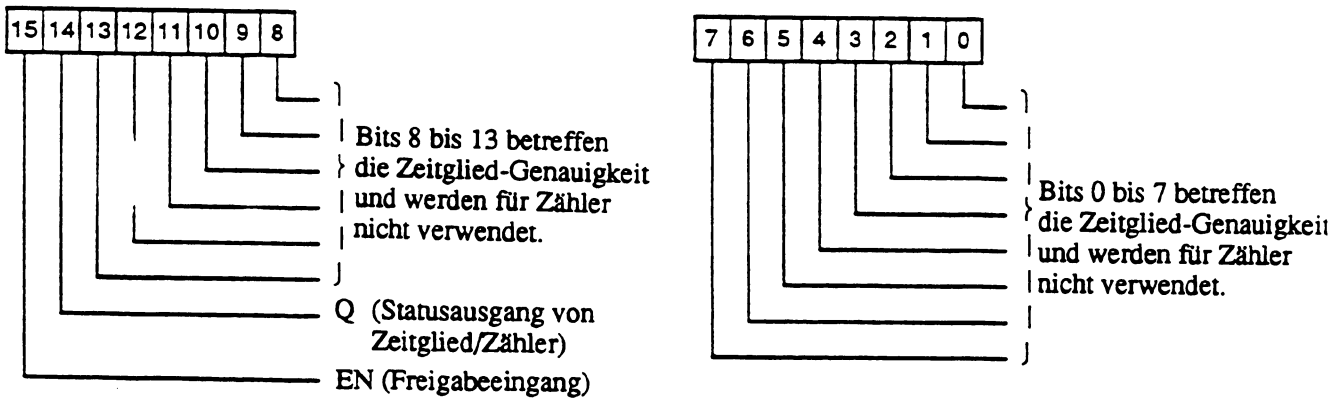
Jedes Zeitglied und jeder Zähler belegt im %P-, %L- oder %R-Speicher drei Worte (Register), um die folgenden Daten abzuspeichern:

Istwert (CV)	Wort 1
Sollwert (PV)	Wort 2
Steuerwort	Wort 3

Bei der Eingabe eines Zeitgliedes oder Zählers müssen Sie immer eine Anfangsadresse für diese drei Worte (Register) unmittelbar unterhalb der Funktionsgraphik eingeben. Zum Beispiel:



Im Steuerwort werden die Zustände des Boolschen Ein- und Ausgangs des zugehörigen Funktionsblocks in folgendem Format abgelegt:



ONDTR

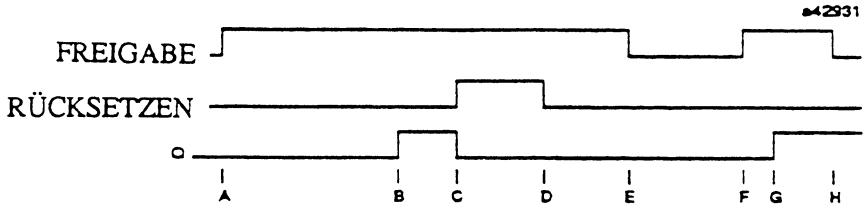
Ein remanentes Zeitglied zur Einschaltverzögerung (ONDTR) inkrementiert seinen Wert, solange es Stromfluß empfängt, und hält den Wert, wenn der Stromfluß stoppt. Die Zeitzählung erfolgt dabei in Schritten von Sekunden (Voreinstellung), 1/10 oder 1/100 Sekunde in einem Bereich zwischen 0 und 32.767 Zeiteinheiten. Dieses Zeitglied ist nullspannungssicher, beim Einschalten wird es nicht automatisch initialisiert.

Die ONDTR-Funktion beginnt mit der Zeitzählung (Istwert CV), wenn sie erstmals Stromfluß empfängt. Der Istwert wird dann aktualisiert, wenn das Zeitglied im Kontaktplanprogramm angetroffen wird.

HINWEIS

Wird das gleiche Zeitglied mit der gleichen Referenzadresse an mehreren Stellen während eines CPU-Zyklus freigegeben, dann werden die Istwerte des Zeitgliedes nochmals mit der vorhergehenden Zykluszeit aktualisiert.

Der Ausgang Q wird durchgeschaltet, wenn der Istwert (CV) gleich dem Sollwert (PV) ist. Der Istwert im Zeitglied wird bis zum Maximalwert erhöht, solange der Stromfluß zu dem Zeitglied besteht.



A = Das Freigabesignal (ENABLE) wird aktiv, das Zeitglied beginnt mit der Zeitzählung.

B = Der Istwert (CV) erreicht den Sollwert (PV); der Ausgang Q wird durchgeschaltet.

C = Das Rücksetzsignal (RESET) wird aktiv; Q wird abgeschaltet, der Istwert rückgesetzt.

D = Das Rücksetzsignal (RESET) wird zurückgenommen; das Zeitglied beginnt wieder mit der Zählung.

E = Das Freigabesignal (ENABLE) wird zurückgenommen; die Zeitzählung wird angehalten. Der Istwert bleibt erhalten.

F = Das Freigabesignal (ENABLE) wird wieder aktiv; das Zeitglied fährt mit der Zeitzählung fort.

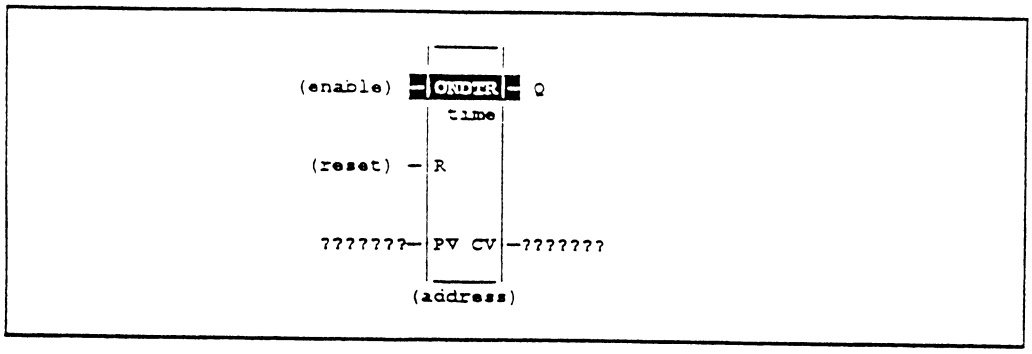
G = Der Istwert (CV) erreicht den Sollwert (PV); der Ausgang Q wird durchgeschaltet. Das Zeitglied fährt mit der Zeitzählung solange fort, bis entweder das Freigabesignal zurückgenommen wird, RESET aktiv wird oder der Istwert den Maximalwert erreicht.

H = Das Freigabesignal (ENABLE) wird zurückgenommen; die Zeitzählung wird angehalten.

Wird der Stromfluß zum Zeitglied unterbrochen, dann wird die Erhöhung des Istwerts angehalten und der *Istwert bleibt erhalten.* Die Erhöhung des Istwerts wird bei diesem erhaltenen Wert fortgesetzt, wenn der Stromfluß zum Zeitglied wieder einsetzt. Bei Stromfluß zum Rücksetzeingang (R) wird der Istwert auf Null gesetzt und der Ausgang wird durchgeschaltet.

Wird die ONDTR-Funktion in einem Programmblock verwendet, der nicht bei jedem Zyklus aufgerufen wird, dann zählt das Zeitglied zwischen den Aufrufen weiter, sofern es nicht rückgesetzt wird. Dies bedeutet, daß es wie ein Zeitglied in einem Programm mit einer viel langsameren Zykluszeit als das Zeitglied im Hauptprogramm arbeitet. In Programmblöcken, die für lange Zeit inaktiv sind, muß das Zeitglied so programmiert werden, daß es diese Aufholfunkion berücksichtigt.

Wird zum Beispiel ein Programmblock rückgesetzt und über einen Zeitraum von 4 Minuten nicht aufgerufen, dann sind beim Aufruf des Programmblocks bereits vier Minuten Zeit aufgelaufen. Das Zeitglied wird mit diesem Zeitwert belegt, wenn es nicht zuerst rückgesetzt wird.



Parameter:

Parameter	Beschreibung
address	Dieser Parameter gibt die Adresse von Istwert, Sollwert und Steuerwort des Zeitglieds an.
enable	Der Istwert des Zeitglieds wird erhöht, wenn ONDTR freigegeben ist.
R	Der Istwert wird auf Null gesetzt, wenn dieser Eingang Stromfluß erhält. Ist PV eine Registerreferenz (%R), dann wird der PV-Parameter als zweites Wort des Adreßparameters angegeben. Ein Adreßparameter %R00001 würde z.B. %R00002 als PV-Parameter verwenden.
PV	Der Wert von PV wird in den Sollwert des Zeitglieds kopiert, wenn das Zeitglied freigegeben oder rückgesetzt wird.
Q	Der Ausgang Q wird durchgeschaltet, wenn der Istwert größer als oder gleich dem Sollwert ist.
CV	CV enthält den Istwert des Zeitglieds. Dieser Wert liegt im Bereich zwischen 0 und 32.767 Zeiteinheiten.

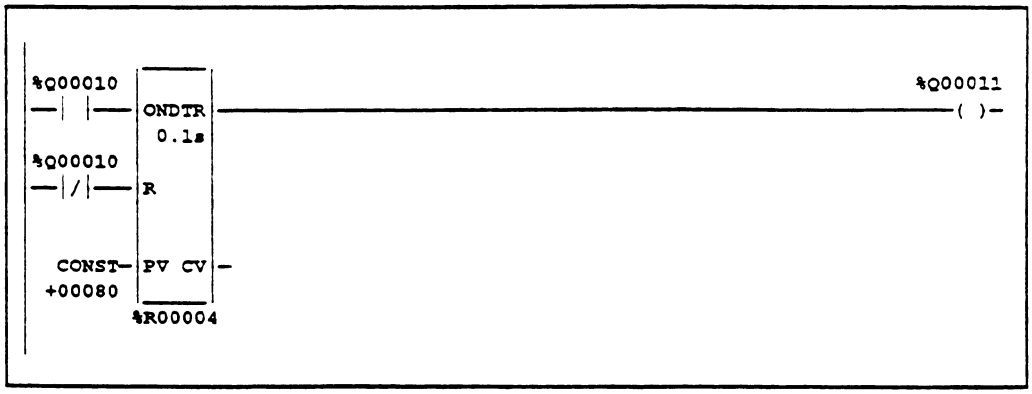
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•	•	•				
enable	•													
R	•													
PV	•	•	•	•	•		•	•	•	•	•	•	•	•
Q	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		•

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Im folgenden Beispiel wird ein remanentes Zeitglied zur Einschaltverzögerung dazu verwendet, ein Signal (%Q00011) zu erzeugen, das 8,0 Sekunden nach Einschalten von %Q00010 aktiv wird und das abgeschaltet wird, wenn %Q00010 abschaltet.



OFDT

Das Zeitglied zur Ausschaltverzögerung (OFDT) inkrementiert seinen Wert, solange es keinen Stromfluß empfängt, und setzt den Wert auf Null, wenn der Stromfluß einsetzt. Die Zeitzählung erfolgt dabei in Schritten von Sekunden (Voreinstellung), 1/10 oder 1/100 Sekunde in einem Bereich zwischen 0 und 32.767 Zeiteinheiten.

Die OFDT-Funktion schaltet durch und setzt den Istwert (CV) auf Null, wenn sie Stromfluß empfängt. Der Ausgang bleibt solange durchgeschaltet, wie die Funktion Stromfluß empfängt. Stoppt der Stromfluß von links zur Funktion, dann gibt die Funktion weiterhin Stromfluß nach rechts weiter und das Zeitglied beginnt mit der Zeitzählung in CV.

HINWEIS

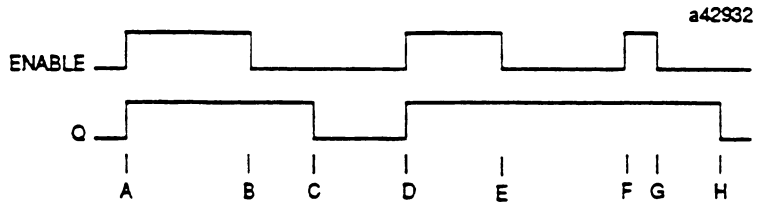
Wird das gleiche Zeitglied mit der gleichen Referenzadresse an mehreren Stellen während eines CPU-Zyklus freigegeben, dann werden die Istwerte des Zeitgliedes mit der vorhergehenden Zykluszeit aktualisiert.

Die OFTD-Funktion schaltet nicht durch, wenn der Sollwert Null oder negativ ist.

Bei jedem Aufruf der Funktion, bei dem die Freigabelogik "0" ist, wird der Istwert (CV) aktualisiert und gibt dann die Zeitspanne seit dem Abschalten des Zeitglieds an. Sind Istwert (CV) und Sollwert (PV) gleich, dann unterbricht die Funktion den Stromfluß nach rechts. Der Istwert (CV) überschreitet niemals den Sollwert (PV).

Empfängt die Funktion wieder Stromfluß, dann wird der Istwert auf Null gesetzt.

Der Zustand der OFDT-Funktion ist nullspannungssicher, beim Einschalten wird sie nicht automatisch initialisiert.



A = Das Freigabesignal (ENABLE) und Q werden beide aktiv.

B = Das Freigabesignal (ENABLE) wird zurückgenommen; das Zeitglied beginnt mit der Zeitzählung.

C = Der Istwert (CV) erreicht den Sollwert (PV); der Ausgang Q wird abgeschaltet und die Zeitzählung wird angehalten.

D = Das Freigabesignal (ENABLE) wird wieder aktiv; der Istwert wird rückgesetzt (CV=0).

E = Das Freigabesignal (ENABLE) wird zurückgenommen; das Zeitglied beginnt mit der Zeitzählung.

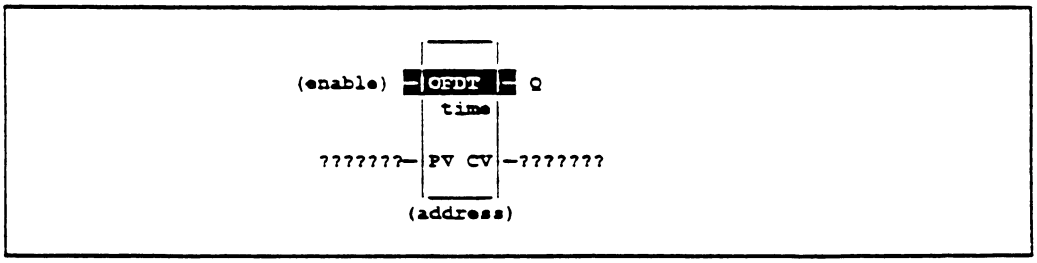
F = Das Freigabesignal (ENABLE) wird aktiv; der Istwert wird rückgesetzt (CV=0).

G = Das Freigabesignal (ENABLE) wird zurückgenommen; das Zeitglied beginnt mit der Zeitzählung.

H = Der Istwert (CV) erreicht den Sollwert (PV); der Ausgang Q wird abgeschaltet und die Zeitzählung wird angehalten.

Wird die OFDT-Funktion in einen Programmblock verwendet, der nicht bei jedem Zyklus aufgerufen wird, dann zählt das Zeitglied zwischen den Aufrufen weiter, sofern es nicht rückgesetzt wird. Dies bedeutet, daß es wie ein Zeitglied in einem Programm mit einer viel langsameren Zykluszeit als das Zeitglied im Hauptprogramm arbeitet. In Programmblöcken, die für lange Zeit inaktiv sind, muß das Zeitglied so programmiert werden, daß es diese Aufholfunkion berücksichtigt.

Wird zum Beispiel ein Programmblock rückgesetzt und über einen Zeitraum von 4 Minuten nicht aufgerufen, dann sind beim Aufruf des Programmblocks bereits vier Minuten Zeit aufgelaufen. Das Zeitglied wird mit diesem Zeitwert belegt, wenn es nicht zuerst rückgesetzt wird.



Parameter:

Parameter	Beschreibung
address	Dieser Parameter gibt die Adresse von Istwert, Sollwert und Steuerwort des Zeitglieds an.
enable	Der Istwert des Zeitglieds wird erhöht, wenn OFDT freigegeben ist.
PV	Der Wert von PV wird in den Sollwert des Zeitglieds kopiert, wenn das Zeitglied freigegeben oder rückgesetzt wird. Ist PV eine Registerreferenz (%R), dann wird der PV-Parameter als zweites Wort des Adreßparameters angegeben. Ein Adreßparameter %R00001 würde z.B. %R00002 als PV-Parameter verwenden.
Q	Der Ausgang Q wird durchgeschaltet, wenn der Istwert größer als oder gleich dem Sollwert ist.
CV	CV enthält den Istwert des Zeitglieds. Dieser Wert liegt im Bereich zwischen 0 und 32.767 Zeiteinheiten.

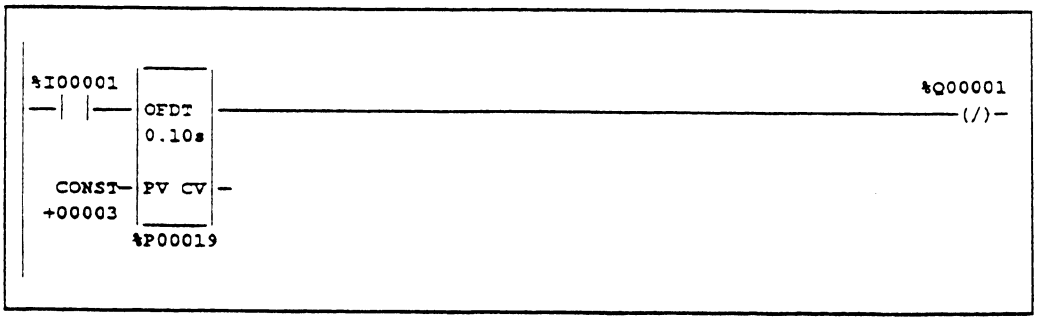
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•	•	•				
enable	•													
PV	•	•	•	•	•		•	•	•	•	•	•	•	•
Q	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		•

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Im folgenden Beispiel wird eine OFDT-Funktion dazu verwendet, einen Ausgang (%Q00001) jedesmal dann abzuschalten, wenn ein Eingang (%I00001) aktiv wird. Der Ausgang wird 0,3 Sekunden nach Abschalten des Eingangs wieder eingeschaltet.



TMR

Das einfache Zeitglied zur Einschaltverzögerung (TMR) inkrementiert seinen Wert, solange es Stromfluß empfängt, und setzt den Wert auf Null, wenn der Stromfluß stoppt. Die Zeitzählung erfolgt dabei in Schritten von Sekunden (Voreinstellung), 1/10- oder 1/100-Sekunden in einem Bereich zwischen 0 und 32.767 Zeiteinheiten.

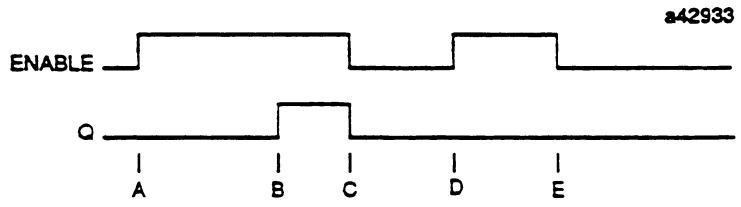
Die TMR-Funktion beginnt mit der Zeitzählung (Istwert), wenn sie Stromfluß empfängt. Der Istwert, der in CV gespeichert wird, wird kontinuierlich aktualisiert, wenn das Zeitglied im Kontaktplanprogramm angetroffen wird, und gibt so die gesamte Zeitdauer an, die seit dem letzten Rücksetzen verstrichen ist.

HINWEIS

Wird das gleiche Zeitglied mit der gleichen Referenzadresse an mehreren Stellen während eines CPU-Zyklus freigegeben, dann werden die Istwerte des Zeitgliedes mit der vorhergehenden Zykluszeit aktualisiert.

Diese Aktualisierung dauert solange, wie die Freigabelogik "wahr" bleibt. Die Funktion schaltet den Stromfluß nach rechts durch, wenn der Istwert (CV) den Sollwert (PV) erreicht hat. Das Zeitglied erhöht den Istwert solange, bis der Maximalwert erreicht ist. Wechselt der Freigabeparameter von "wahr" nach "falsch", dann stoppt das Zeitglied und der Istwert wird auf Null rückgesetzt.

Der Zustand der TMR-Funktion ist nullspannungssicher, beim Einschalten wird sie nicht automatisch initialisiert.



A = Das Freigabesignal (ENABLE) wird aktiv, das Zeitglied beginnt mit der Zeitzählung.

B = Der Istwert (CV) erreicht den Sollwert (PV); der Ausgang Q wird durchgeschaltet. Das Zeitglied fährt mit der Zeitzählung fort.

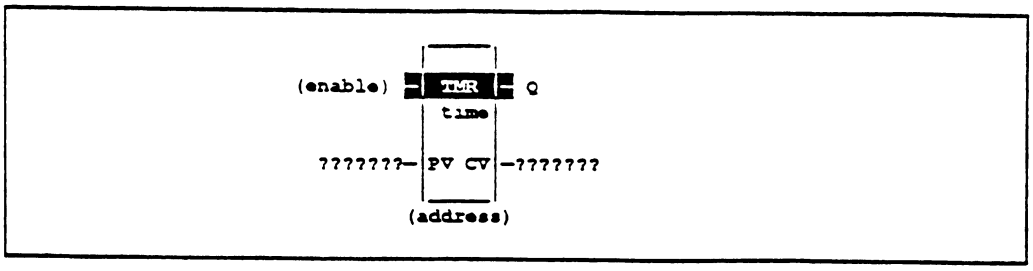
C = Das Freigabesignal (ENABLE) wird zurückgenommen; die Zeitzählung wird angehalten. Der Istwert (CV) wird gelöscht.

D = Das Freigabesignal (ENABLE) wird aktiv, das Zeitglied beginnt mit der Zeitzählung.

E = Das Freigabesignal (ENABLE) wird zurückgenommen ehe der Istwert (CV) den Sollwert (PV) erreicht. Q bleibt abgeschaltet und die Zeitzählung wird angehalten.

Wird die TMR-Funktion in einen Programmblock verwendet, der nicht bei jedem Zyklus aufgerufen wird, dann zählt das Zeitglied zwischen den Aufrufen weiter, sofern es nicht rückgesetzt wird. Dies bedeutet, daß es wie ein Zeitglied in einem Programm mit einer viel langsameren Zykluszeit als das Zeitglied im Hauptprogramm arbeitet. In Programmblöcken, die für lange Zeit inaktiv sind, muß das Zeitglied so programmiert werden, daß es diese Aufholfunkion berücksichtigt.

Wird zum Beispiel ein Programmblock rückgesetzt und über einen Zeitraum von 4 Minuten nicht aufgerufen, dann sind beim Aufruf des Programmblocks bereits vier Minuten Zeit aufgelaufen. Das Zeitglied wird mit diesem Zeitwert belegt, wenn es nicht zuerst rückgesetzt wird.



Parameter:

Parameter	Beschreibung
address	Dieser Parameter gibt die Adresse von Istwert, Sollwert und Steuerwort des Zeitglieds an.
enable	Wenn TMR freigegeben ist, dann wird der Istwert des Zeitglieds erhöht. Ist TMR nicht freigegeben, dann wird der Istwert auf Null gesetzt und Q abgeschaltet.
PV	Der Wert von PV wird in den Sollwert des Zeitglieds kopiert, wenn das Zeitglied freigegeben oder rückgesetzt wird. Ist PV eine Registerreferenz (%R), dann wird der PV-Parameter als zweites Wort des Adreßparameters angegeben. Ein Adreßparameter %R00001 würde z.B. %R00002 als PV-Parameter verwenden.
Q	Der Ausgang Q wird durchgeschaltet, wenn TMR freigegeben und der Istwert größer als oder gleich dem Sollwert ist.
CV	CV enthält den Istwert des Zeitglieds. Dieser Wert liegt im Bereich zwischen 0 und 32.767 Zeiteinheiten.

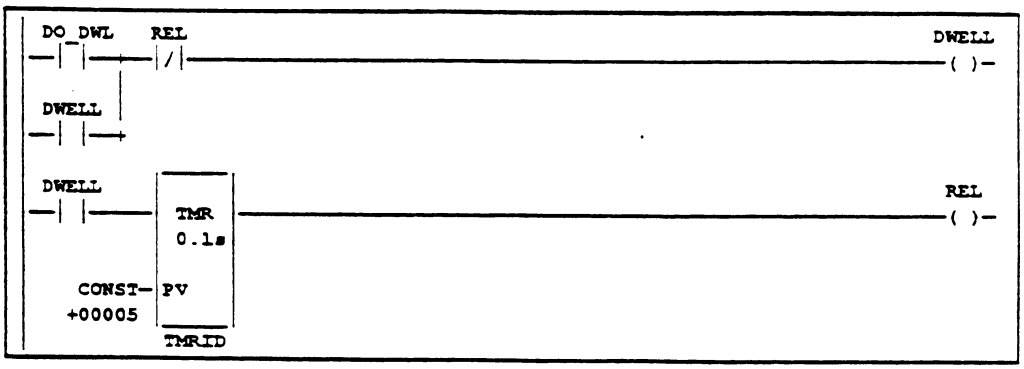
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•	•	•				
enable	•													
PV	•	•	•	•	•		•	•	•	•	•	•	•	•
Q	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		•

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

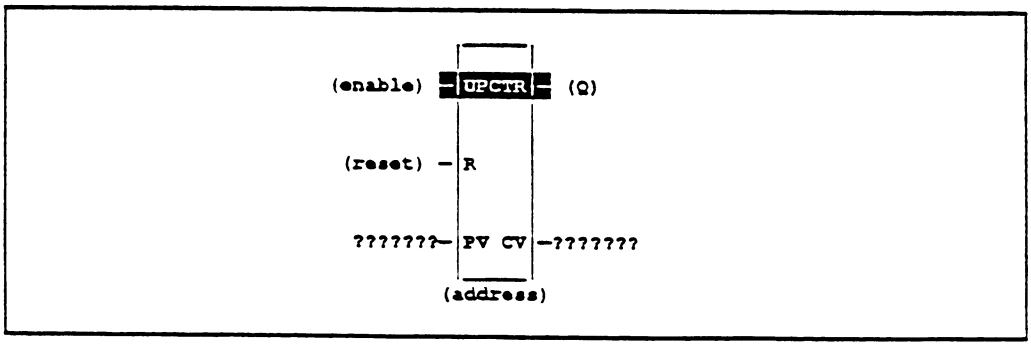
In dem folgenden Beispiel wird ein Zeitglied zur Einschaltverzögerung TMRID dazu verwendet, die Einschaltdauer einer Spule DWELL zu steuern. Die Spule DWELL wird durchgeschaltet, wenn der (momentane) Schließerkontakt DO_DWL geschlossen ist. Der Haltekontakt der Spule DWELL hält die Spule DWELL durchgeschaltet (wenn der Kontakt DO_DWL losgelassen wird) und startet das Zeitglied TMRID. Hat TMRID seinen Sollwert (½ Sekunde) erreicht, schaltet die Spule REL durch und unterbricht den Haltezustand der Spule DWELL. Der Kontakt DWELL unterbricht den Stromfluß zu TMRID, setzt dessen Istwert zurück und schaltet die Spule REL ab. Die Schaltung ist dann für die nächste Aktivierung des Kontakts DO_DWL bereit



UPCTR

Mit der Funktion des Aufwärtszählers (UPCTR) wird bis zu einem vorgegebenen Wert vorwärts gezählt. Der Wertebereich beträgt dabei zwischen 0 und 32.767. Wird der Rücksetzeingang des Aufwärtszählers aktiviert, dann wird der Istwert auf Null gesetzt. Wechselt der Zähleingang von "falsch" auf "wahr", dann wird der Istwert um eins erhöht. der Istwert kann über den Sollwert (PV) hinaus erhöht werden. Der Ausgang wird durchgeschaltet, wenn Istwert und Sollwert gleich sind oder der Istwert größer als der Sollwert ist.

Der Zustand der UPCTR-Funktion ist nullspannungssicher, beim Einschalten wird die Funktion nicht automatisch initialisiert.



Parameter:

Parameter	Beschreibung
address	Dieser Parameter gibt die Adresse von Istwert, Sollwert und Steuerwort des Zählers an.
enable	Bei einer positiven Anderung des Freigabeeingangs wird der Istwert um eins erhöht.
R	Der Istwert wird auf Null rückgesetzt, wenn R Stromfluß empfängt.
PV	Der Wert von PV wird in den Sollwert des Zählers kopiert, wenn der Zählers freigegeben oder rückgesetzt wird. Ist PV eine Registerreferenz (%R), dann wird der PV-Parameter als zweites Wort des Adreßparameters angegeben. Ein Adreßparameter %R00001 würde z.B. %R00002 als PV-Parameter verwenden.
Q	Der Ausgang Q wird durchgeschaltet, wenn der Istwert größer als oder gleich dem Sollwert ist.
CV	CV enthält den Istwert des Zeitglieds. Dieser Wert liegt im Bereich zwischen 0 und 32.767 Zeiteinheiten.

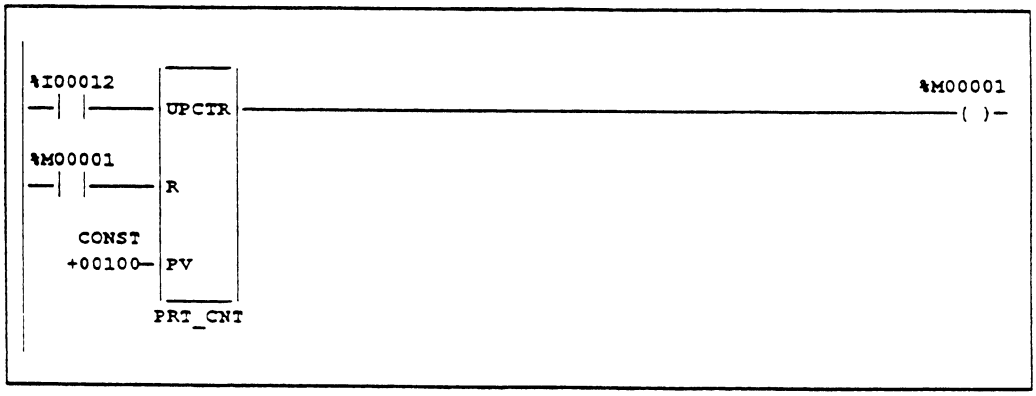
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•	•	•				
enable	•													
R	•													
PV	•	•	•	•	•		•	•	•	•	•	•	•	•
Q	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		•

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

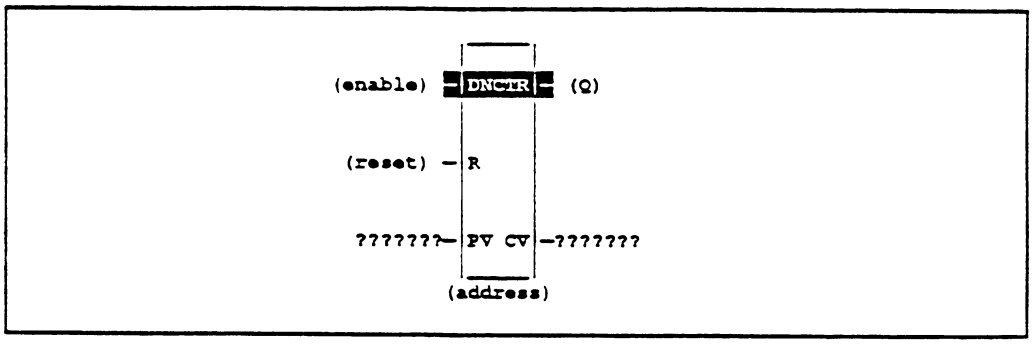
In dem folgenden Beispiel wird der Aufwärtszähler PRT-CNT jedesmal um eins weitergeschaltet, wenn der Eingang %I00012 von "falsch" auf "wahr" wechselt. Der interne Merker %M00001 wird durchgeschaltet, wenn 100 Einheiten gezählt wurden. Der Zählwert wird auf Null gesetzt, wenn %M00001 "wahr" wird.



DNCTR

Mit der Funktion des Abwärtszählers (DNCTR) wird von einem vorgegebenen Wert aus rückwärts auf Null gezählt. Der maximale Vorgabewert beträgt dabei 32.767 Zählwerte. Wird der Rücksetzeingang des Abwärtszählers aktiviert, dann wird der Istwert auf den Sollwert (PV) gesetzt. Wechselt der Freigabeeingang von "falsch" auf "wahr", dann wird der Istwert um eins erniedrigt. Der Ausgang wird durchgeschaltet, wenn der Istwert gleich oder kleiner Null ist.

Der Istwert der DNCTR-Funktion ist nullspannungssicher, beim Einschalten wird die Funktion nicht automatisch initialisiert.



Parameter:

Parameter	Beschreibung
address	Dieser Parameter gibt die Adresse von Istwert, Sollwert und Steuerwort des Zählers an.
enable	Bei einer positiven Anderung des Freigabeeingangs wird der Istwert um eins erniedrigt.
R	Der Istwert wird auf den Sollwert rückgesetzt, wenn R Stromfluß empfängt.
PV	Der Wert von PV wird in den Sollwert des Zählers kopiert, wenn der Zählers freigegeben oder rückgesetzt wird. Ist PV eine Registerreferenz (%R), dann wird der PV-Parameter als zweites Wort des Adreßparameters angegeben. Ein Adreßparameter %R00001 würde z.B. %R00002 als PV-Parameter verwenden.
Q	Der Ausgang Q wird durchgeschaltet, wenn der Istwert kleiner als oder gleich dem Sollwert ist.
CV	CV enthält den Istwert des Zeitglieds. Dieser Wert liegt im Bereich zwischen 0 und 32.767 Zeiteinheiten.

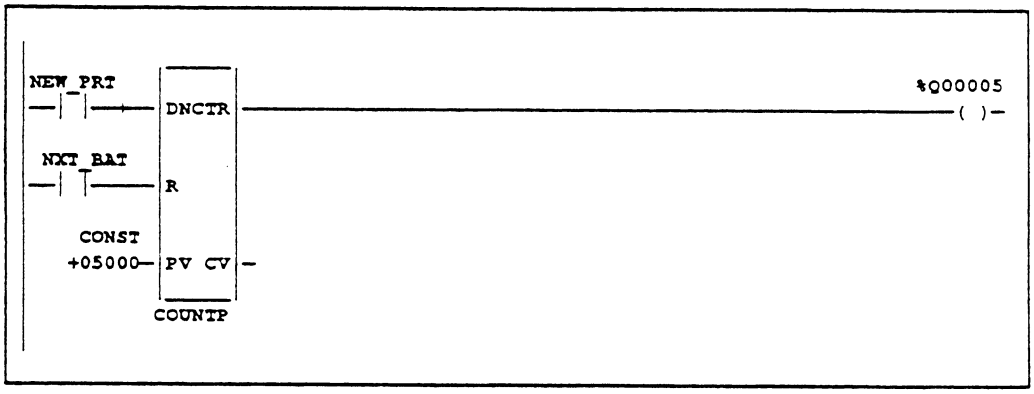
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•	•	•				
enable	•													
R	•													
PV	•	•	•	•	•		•	•	•	•	•	•	•	•
Q	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		•

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

In dem folgenden Beispiel zählt der mit COUNTP bezeichnete Abwärtszähler 5000 neue Teile, ehe der Ausgang %Q00005 gesetzt wird.



KAPITEL 5

In diesem Kapitel werden die arithmetischen Funktionen von Logicmaster 90-70 beschrieben.

Abkürzung	Funktion	Beschreibung
ADD	Addition	Addition zweier Zahlen
SUB	Subtraktion	Subtraktion einer Zahl von einer anderen
MUL	Multiplikation	Multiplikation zweier Zahlen
DIV	Division	Division einer Zahl durch eine andere, Ergebnis ist Quotient
MOD	Modulo	Division einer Zahl durch eine andere, Ergebnis ist Rest
SQRT	Quadratwurzel	Zieht die Quadratwurzel aus einer ganzen oder reellen Zahl.
ABS	Absolutwert	Findet den Absolutwert einer ganzen, doppeltgenauen ganzen oder reellen Zahl.
SIN, COS, TAN, ASIN, ACOS, ATAN	Trigonometrische Funktionen	Führt die entsprechende Funktion mit dem am Eingang IN angegebenen reellen Wert durch.
LOG, LN, EXP, EXPT	Logarithmische und Exponentialfunktionen	Führt die entsprechende Funktion mit dem am Eingang IN angegebenen reellen Wert durch.
RAD, DEG	Bogenmaß-Konvertierung	Führt die entsprechende Funktion mit dem am Eingang IN angegebenen reellen Wert durch.

MATH (ADD, SUB, MUL, DIV)

Zu den arithmetischen MATH-Funktionen gehören Addition, Subtraktion, Multiplikation und Division. Empfängt die Funktion Stromfluß, dann wird die entsprechende MATH-Funktion mit den an I1 und I2 anliegenden Werten ausgeführt, die beide vom gleichen Datentyp sein müssen. Der Ausgangswert Q ist vom gleichen Datentyp wie I1 und I2. Werden ganze Zahlen mit Vorzeichen oder doppelter Genauigkeit verwendet, dann hängt das Vorzeichen des Ergebnisses von DIV und MUL von den Vorzeichen von I1 und I2 ab.

Die MATH-Funktionen können mit den folgenden Datentypen verwendet werden:

Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
UINT	Ganze Zahl ohne Vorzeichen
DINT	Doppeltgenaue ganze Zahl
REAL	Gleitpunkt
MIXED	Gemischt, nur für MUL und DIV.

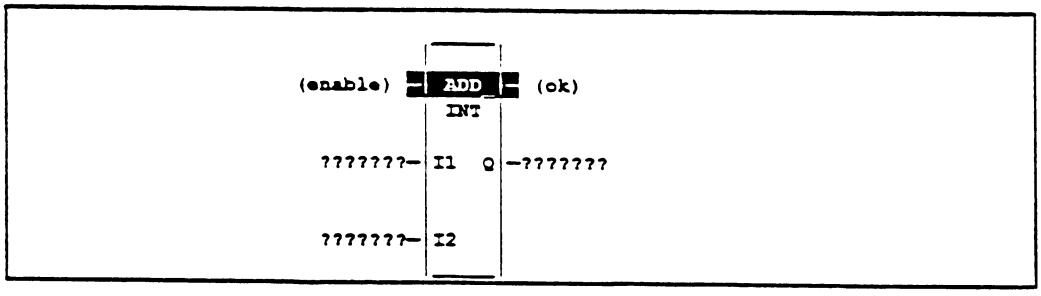
Der Standard-Datentyp "ganzzahlig mit Vorzeichen" kann nach Anwahl der Funktion verändert werden. Weitere Informationen über Datentypen finden Sie in Kapitel 2 dieses Handbuches.

Ergibt die Operation bei INT- oder DINT-Operanden einen Überlauf, dann wird die Ausgangsvariable auf den größten für diesen Datentyp möglichen Wert eingestellt. Ergibt die Operation bei UINT-Operanden einen Überlauf, dann wird die Ausgangsvariable auf den Überlaufwert eingestellt. Bei vorzeichenbehafteten Zahlen zeigt das Vorzeichen die Richtung des Überlaufs an.

Ergibt das Ergebnis keinen Überlauf, dann wird der OK-Ausgang auf "wahr" gesetzt, sofern keine der folgenden unzulässigen REAL-Operationen auftritt:

- Bei ADD: $+\infty + -\infty$
- Bei SUB: $\pm\infty - \pm\infty$
- Bei MUL: $0 \times \infty$
- Bei DIV: $0 / 0$
- Bei DIV: ∞ / ∞
- I1 oder I2 ist keine Zahl.

In jedem dieser Fälle wird OK auf "falsch" gesetzt.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für den ersten in der Operation verwendeten Wert.
I2	I2 enthält eine Konstante oder Referenz für den zweiten in der Operation verwendeten Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion ohne Uberlauf und fehlerfrei ausgeführt wurde und für I1 bzw. I2 jeweils eine Zahl eingegeben wurde.
Q	Der Ausgang Q enthält das Ergebnis der Operation.

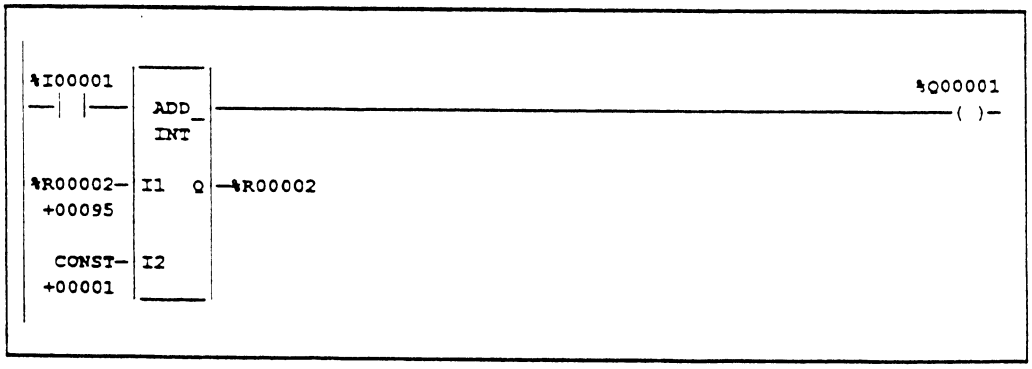
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o		o	•	•	•	•	•	•	
I2	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT- oder UINT-Daten zulässig.

Beispiel:

In dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, der ganzzahlige Inhalt von %R00002 um eins erhöht. Die Spule %Q00001 wird immer dann durchgeschaltet, wenn die Addition fehlerfrei ausgeführt wurde.



MOD (INT, DINT, UINT)

Mit der Modulofunktion wird ein Wert durch einen anderen Wert vom gleichen Typ dividiert. Das Ergebnis ist ein Rest. Bei Verwendung von vorzeichenbehafteten oder doppeltgenauen ganzen Zahlen hängt das Vorzeichen des Ergebnisses von dem Vorzeichen von I1 ab.

Die Modulofunktion kann für folgende Datentypen verwendet werden:

Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
UINT	Ganze Zahl ohne Vorzeichen
DINT	Doppeltgenaue ganze Zahl

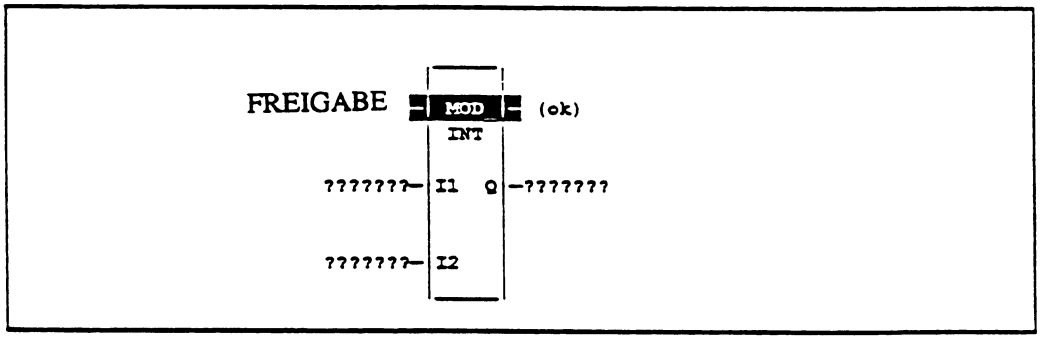
Der Standard-Datentyp "ganzzahlig mit Vorzeichen" kann nach Anwahl der Funktion verändert werden. Weitere Informationen über Datentypen finden Sie in Kapitel 2 dieses Handbuches.

Empfängt die Funktion Stromfluß, dann dividiert sie den Wert I1 durch den Wert I2. Beide Werte müssen vom gleichen Datentyp sein. Der Ausgangswert Q berechnet sich zu:

$$Q = I1 - ((I1 \text{ DIV } I2) * I2)$$

Die Division (DIV) ergibt dabei eine ganze Zahl. Der Ausgangswert Q ist vom gleichen Datentyp wie I1 und I2.

Der OK-Ausgang ist immer "wahr", wenn die Funktion Stromfluß enthält und nicht versucht wurde, durch Null zu dividieren. In einem solchen Fall wird der OK-Ausgang auf "falsch" gesetzt.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für den Dividenden.
I2	I2 enthält eine Konstante oder Referenz für den Divisor.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde.
Q	Der Ausgang Q enthält den Rest der Division von I1 durch I2.

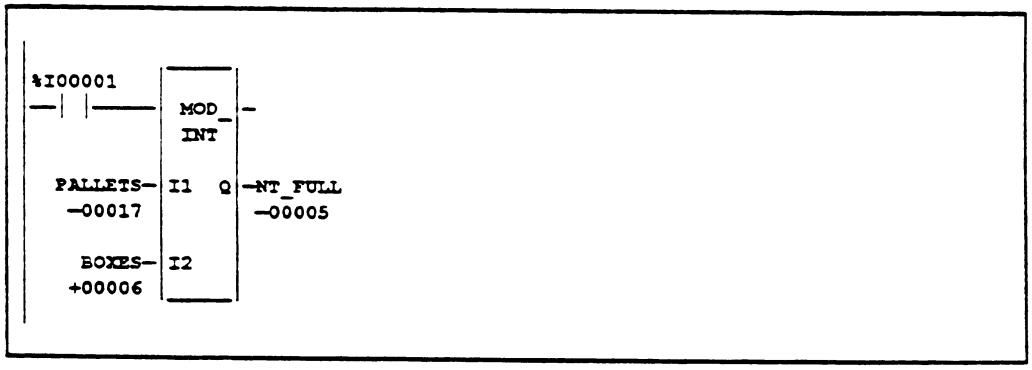
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o		o	•	•	•	•	•	•	
I2	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT- oder UINT-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, der Rest der ganzzahligen Division von PALLETS durch BOXES in NT-FUL eingetragen.



SQRT (INT, DINT, REAL)

Mit der Wurzelfunktion kann die Quadratwurzel eines ganzzahligen Wertes gezogen werden. Erhält die Wurzelfunktion Stromfluß, dann wird der Wert des Ausgangs Q auf den ganzzahligen Wert der Quadratwurzel aus dem Eingangswert IN gesetzt. Der Ausgangswert Q ist vom gleichen Datentyp wie IN.

Die Wurzelfunktion kann für folgende Datentypen verwendet werden:

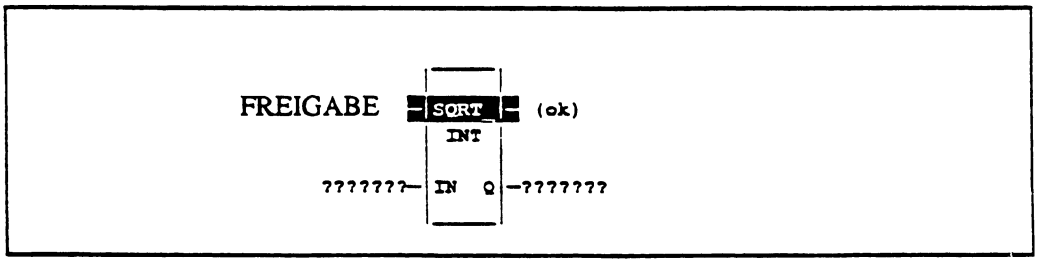
Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
DINT	Doppeltgenaue ganze Zahl
REAL	Gleitpunktzahl

Der Standard-Datentyp "ganzzahlig mit Vorzeichen" kann nach Anwahl der Funktion verändert werden. Weitere Informationen über Datentypen finden Sie in Kapitel 2 dieses Handbuches.

Der OK-Ausgang wird "wahr", wenn die Funktion ohne Überlauf ausgeführt wurde und keine der folgenden unzulässigen REAL-Operationen auftritt:

- IN < 0
- IN ist keine Zahl.

In jedem dieser Fälle wird OK auf "falsch" gesetzt.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Konstante oder Referenz, aus deren Wert die Wurzel gezogen werden soll.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde und für IN eine Zahl eingegeben wurde.
Q	Der Ausgang Q enthält die Quadratwurzel von IN.

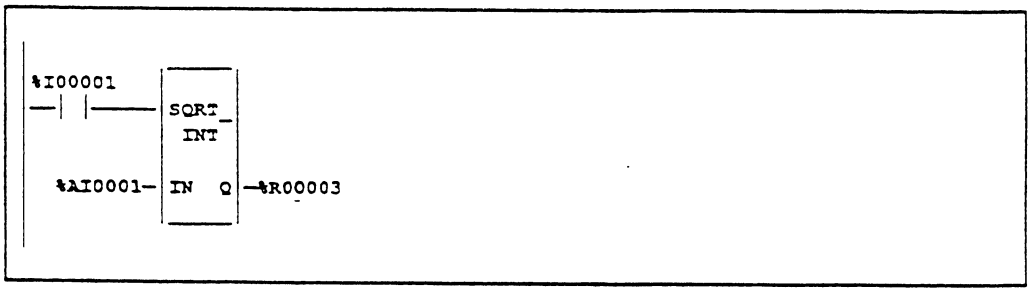
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, die Quadratwurzel der in %AI0001 enthaltenen ganzen Zahl in das Ergebnisregister %R00003 eingetragen.



ABS (INT, DINT, REAL)

Mit der Absolutwertfunktion können sie den Absolutwert einer ganzen Zahl, einer doppeltgenauen ganzen Zahl oder einer reellen Zahl ermitteln. Wenn die Funktion Stromfluß empfängt, dann legt sie den Absolutwert des an Eingang IN anliegenden Wertes im Ausgang Q ab.

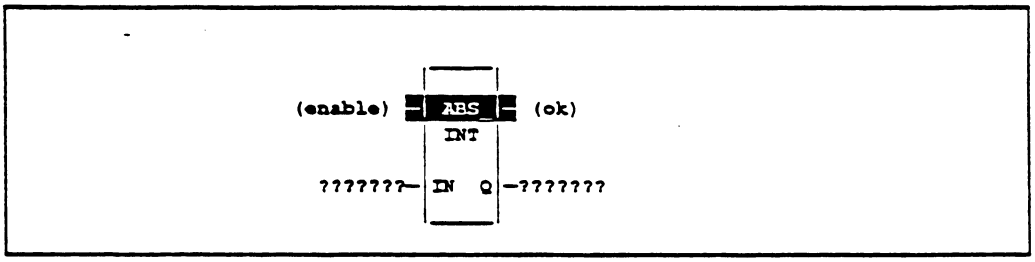
Die Absolutwertfunktion kann für folgende Datentypen verwendet werden:

Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
DINT	Doppeltgenaue ganze Zahl
REAL	Gleitpunktzahl

Der Standard-Datentyp "ganzzahlig mit Vorzeichen" kann nach Anwahl der Funktion verändert werden. Weitere Informationen über Datentypen finden Sie in Kapitel 2 dieses Handbuches.

Der OK-Ausgang wird "wahr", wenn keine der folgenden unzulässigen Bedingungen auftritt:

- Datentyp INT: IN ist MININT.
- Datentyp DINT: IN ist MINDINT.
- Datentyp REAL: IN ist keine Zahl.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält den zu bearbeitenden ganzzahligen oder reellen Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei und ohne Überlauf ausgeführt wurde und für IN eine Zahl eingegeben wurde.
Q	Der Ausgang Q enthält den Absolutwert von IN.

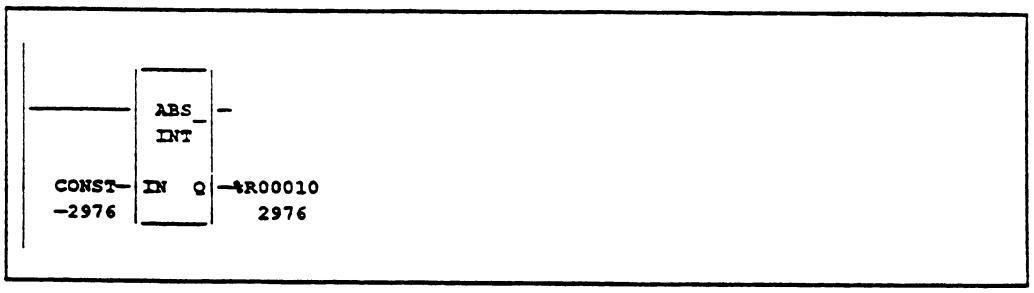
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•	•	

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT-Daten zulässig.

Beispiel:

Beim folgenden Beispiel wird der Absolutwert von -2976 (d.h. 2976) in %R00010 eingetragen.



Trigonometrische Funktionen (SIN, COS, TAN, ASIN, ACOS, ATAN)

Mit den trigonometrischen Funktionen SIN, COS und TAN können Sie die Sinus-, Cosinus- und Tangenswerte des Eingangswertes ermitteln. Erhält eine dieser Funktionen Stromfluß, dann berechnet sie den Sinus-, Cosinus- oder Tangenswert des in Radiant angegebenen Wertes von IN und legt das Ergebnis im Ausgang Q ab. IN und Q sind beides Gleitpunktwerte.

Mit den trigonometrischen Funktionen ASIN, ACOS und ATAN können Sie die inversen Sinus-, Cosinus- und Tangenswerte des Eingangswertes ermitteln. Erhält eine dieser Funktionen Stromfluß, dann berechnet sie den inversen Sinus-, Cosinus- oder Tangenswert des Wertes von IN und legt das Ergebnis in Radiant im Ausgang Q ab. IN und Q sind beides Gleitpunktwerte.

Bei den SIN-, COS- und TAN-Funktionen ist jeweils ein weiter Bereich von Eingangswerten zulässig: $-2^{63} < IN < +2^{63}$ ($2^{63} = 9{,}22 \times 10^{18}$).

Bei ASIN und ACOS ist jeweils nur der Bereich $-1 \leq IN \leq 1$ zulässig. Mit einem zulässigen Wert für den Parameter IN ergibt die Funktion ASIN_REAL den folgenden Wert von Q:

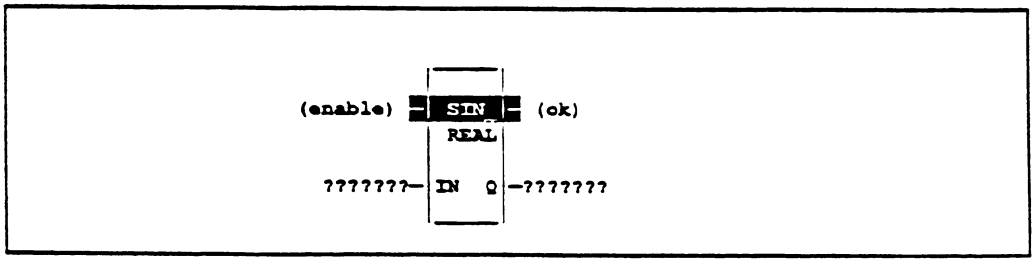
$$ASIN\ (IN) = -\pi/2 \leq Q \leq \pi/2$$

Die Funktion ACOS_REAL ergibt den folgenden Wert von Q:

$$ACOS\ (IN) = 0 \leq Q \leq \pi$$

Die ATAN-Funktion erlaubt den größten Wertebereich für den Eingang: $-\infty \leq IN \leq +\infty$. Mit einem zulässigen Wert für den Parameter IN ergibt die Funktion ATAN_REAL den folgenden Wert von Q:

$$ATAN\ (IN) = -\pi/2 \leq Q \leq \pi/2$$



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält den zu bearbeitenden reellen Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei und ohne Uberlauf ausgeführt wurde und für IN eine Zahl eingegeben wurde.
Q	Der Ausgang Q enthält den trigonometrischen Wert von IN.

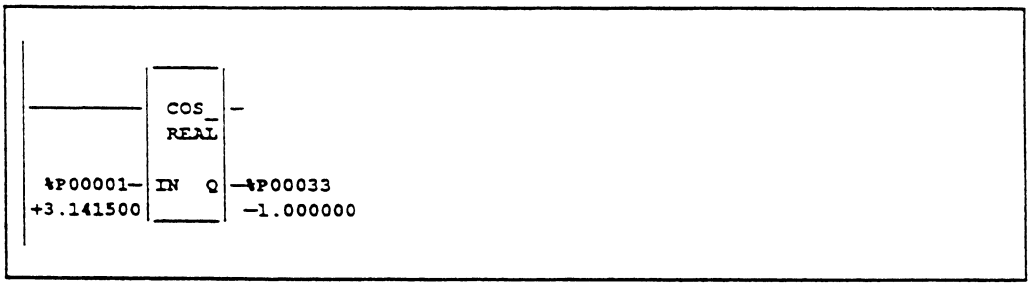
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•							•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Im folgenden Beispiel wird der Cosinus des Wertes aus %P00001 in %P00033 eingetragen.



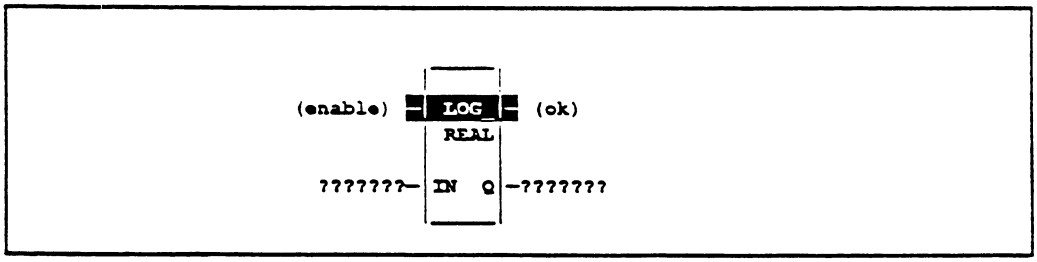
Logarithmische und Exponentialfunktionen (LOG, LN, EXP, EXPT)

Die Funktionen LOG, LN, EXP und EXPT besitzen zwei Eingangsparameter und zwei Ausgangsparameter. Erhält eine dieser Funktionen Stromfluß, dann berechnet sie den entsprechenden Logarithmus bzw. Exponentialwert des reellen Wertes am Eingang IN und legt das Ergebnis in Ausgang Q ab.

- Bei der LOG-Funktion wird der dekadische Logarithmus (Basis 10) des an IN anliegenden Wertes in Q eingetragen.
- Bei der LN-Funktion wird der natürliche Logarithmus (Basis 2) des an IN anliegenden Wertes in Q eingetragen.
- Bei der EXP-Funktion wird *e* zu der durch IN festgelegten Potenz erhoben. Das Ergebnis wird in Q eingetragen.

Die EXPT-Funktion besitzt drei Eingangsparameter und zwei Ausgangsparameter. Erhält die Funktion Stromnfluß, dann wird der Wert von I1 zu der durch I2 festgelegten Potenz erhoben. Das Ergebnis wird in Q eingetragen.

Am OK-Ausgang liegt immer Stromfluß an, solange IN eine Zahl und nicht negativ ist.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält den zu bearbeitenden reellen Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei und ohne Uberlauf ausgeführt wurde und für IN eine Zahl eingegeben wurde, die nicht negativ ist.
Q	Der Ausgang Q enthält den Logarithmus/Exponentialwert von IN.

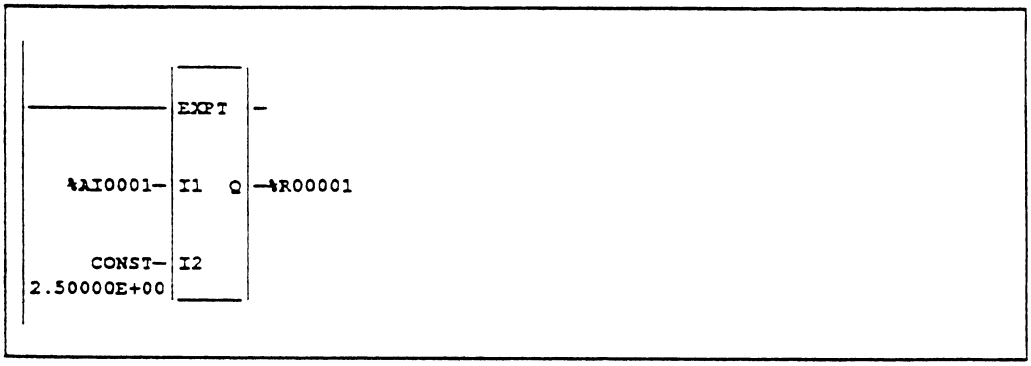
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN *	•							•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
* = Bei der EXPT-Funktion wird Eingang IN durch die Eingangsparameter I1 und I2 ersetzt.
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

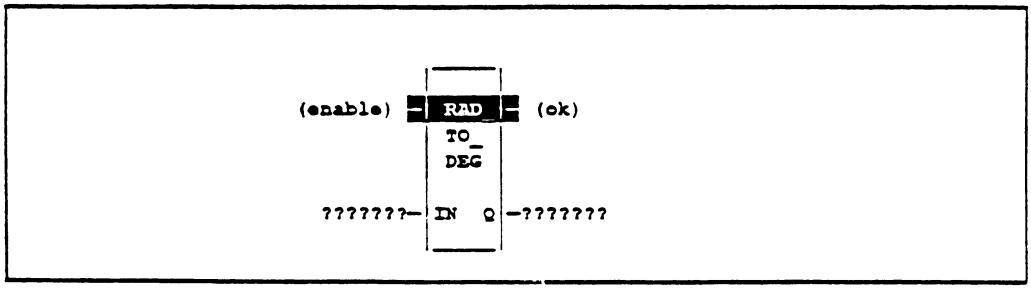
In dem folgenden Beispiel wird der Wert aus %AI0001 zur Potenz von 2,5 erhoben und das Ergebnis wird in %R00001 abgelegt.



Bogenmaß-Konvertierung (RAD, DEG)

Erhält die Funktion Stromfluß, dann wird der am Eingang IN anliegenden reelle Wert entsprechend konvertiert (RAD_TO_DEG oder DEG_TO_RAD). Das Ergebnis wird im Ausgang Q abgelegt.

Am OK-Ausgang liegt Stromfluß an, wenn IN eine Zahl ist.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält den zu bearbeitenden reellen Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion ohne Überlauf ausgeführt wurde und für IN eine Zahl eingegeben wurde.
Q	Der Ausgang Q enthält den konvertierten Wert von IN.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•							•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

KAPITEL 6

Mit relationalen Funktionen werden zwei Zahlen miteinander verglichen. In diesem Kapitel werden die folgenden relationalen Funktionen beschrieben:

Abkürzung	Funktion	Beschreibung
EQ	Gleich	Uberprüft, ob zwei Zahlen gleich sind
NE	Ungleich	Uberprüft, ob zwei Zahlen verschieden sind
GT	Größer als	Uberprüft, ob eine Zahl größer als eine andere ist
GE	Größer/gleich	Uberprüft, ob eine Zahl größer als oder gleich groß wie eine andere ist
LT	Kleiner als	Uberprüft, ob eine Zahl kleiner als eine andere ist
LE	Kleiner/gleich	Uberprüft, ob eine Zahl kleiner als oder gleich groß wie eine andere ist
CMP	Vergleich	Uberprüft, ob eine Zahl kleiner, gleich groß oder größer als eine andere ist

Relationale Funktionen (EQ, NE, GT, GE, LT, LE)

Zu den relationalen Funktionen gehören:

Abkürzung	Funktion	Beschreibung
EQ	Gleich	Überprüft, ob zwei Zahlen gleich sind
NE	Ungleich	Überprüft, ob zwei Zahlen verschieden sind
GT	Größer als	Überprüft, ob eine Zahl größer als eine andere ist
GE	Größer/gleich	Überprüft, ob eine Zahl größer als oder gleich groß wie eine andere ist
LT	Kleiner als	Überprüft, ob eine Zahl kleiner als eine andere ist
LE	Kleiner/gleich	Überprüft, ob eine Zahl kleiner als oder gleich groß wie eine andere ist

Mit relationalen Fuunktionen wird das Verhältnis zweier Werte zueinander bestimmt. Erhält die Funktion Stromfluß, dann vergleicht sie den Wert von I1 mit dem Wert von I2. Beide Werte müssen vom gleichen Datentyp sein.

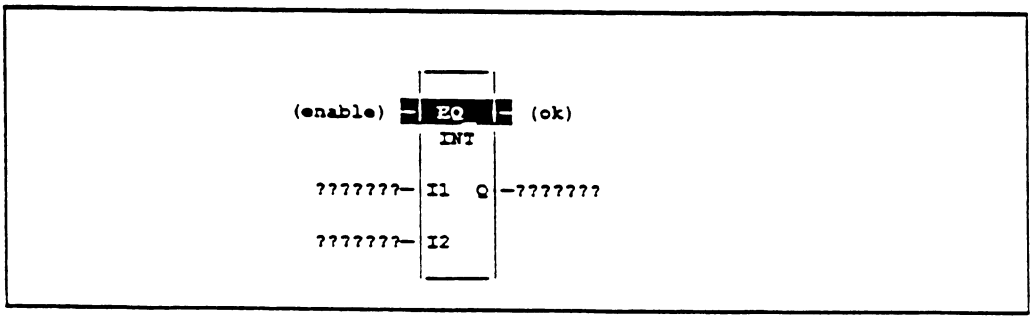
Relationale Funktionen verwenden die folgenden Datentypen:

Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
DINT	Doppeltgenaue ganze Zahl
UINT	Ganze Zahl ohne Vorzeichen
REAL	Gleitpunktzahl

Die Standardbelegung (ganze Zahl mit Vorzeichen) kann nach Anwahl der Funktion verändert werden, wenn ganze Zahlen ohne Vorzeichen oder doppeltgenaue ganze Zahlen miteinander verglichen werden sollen. Wollen Sie Daten mit unterschiedlichen oder anderen Datentypen miteinander vergleichen, dann müssen Sie zunächst die entsprechende Konvertierungsfunktion anwenden (siehe Kapitel 10), um die Daten in einen der ganzzahligen Typen umzuwandeln.

Entsprechen I1 und I2 der angegebenen Relation, dann wird Ausgang Q durchgeschaltet und auf "wahr" (1) gesetzt. Entsprechen I1 und I2 dieser Relation nicht, dann wird Ausgang Q auf "falsch" (0) gesetzt.

Der OK-Ausgang ist durchgeschaltet, wenn die Funktion freigegeben ist und für I1 und I2 eine Zahl eingegeben wurde.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für den ersten zu vergleichenden Wert.
I2	I2 enthält eine Konstante oder Referenz für den zweiten zu vergleichenden Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde und I1 und I2 Zahlen sind.
Q	Ausgang Q wird durchgeschaltet, wenn I1 und I2 der angegebenen Relation entsprechen.

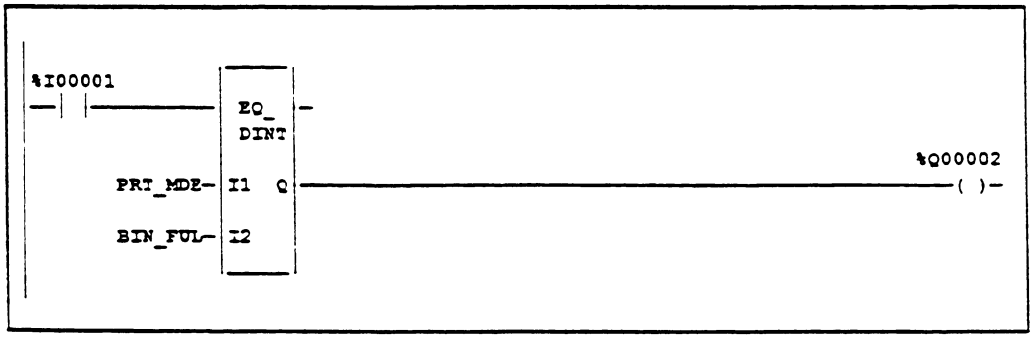
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o		o	•	•	•	•	•	•	
I2	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT- oder UINT-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel werden jedesmal, wenn der Eingang %I00001 gesetzt wird, die beiden doppeltgenauen ganzzahligen Werte PRT_MDE und BIN_FUL miteinander verglichen. Sind die beiden Werte gleich, dann wird Spule %Q00002 durchgeschaltet.



CMP (INT, DINT, UINT, REAL)

Mit der CMP-Funktion wird untersucht, ob ein Wert kleiner, größer oder gleich bezogen auf einen zweiten Wert ist.

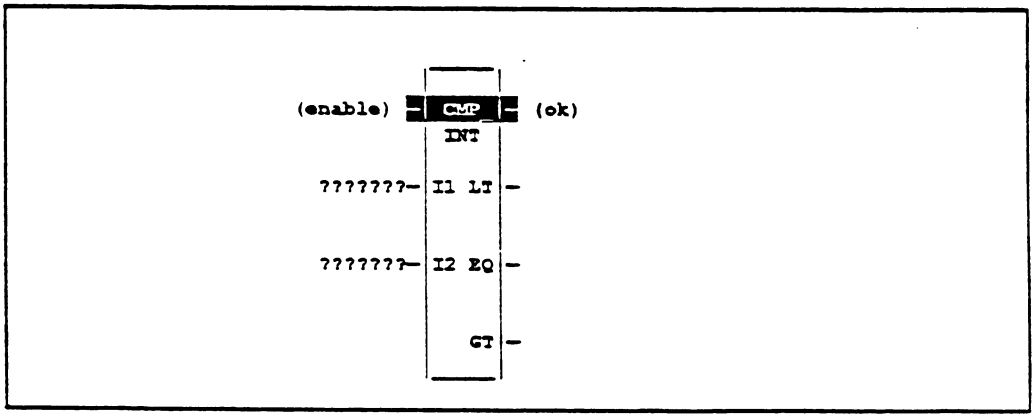
Erhält die Funktion Stromfluß, dann vergleicht sie die beiden Werte I1 und I2 miteinander. Beide Werte müssen vom gleichen Datentyp sein.

Datentyp	Beschreibung
INT	Ganze Zahl mit Vorzeichen
DINT	Doppeltgenaue ganze Zahl
UINT	Ganze Zahl ohne Vorzeichen
REAL	Gleitpunktzahl

Die Standardbelegung (ganze Zahl mit Vorzeichen) kann nach Anwahl der Funktion verändert werden, wenn ganze Zahlen ohne Vorzeichen oder doppeltgenaue ganze Zahlen miteinander verglichen werden sollen. Wollen Sie Daten mit unterschiedlichen oder anderen Datentypen miteinander vergleichen, dann müssen Sie zunächst die entsprechende Konvertierungsfunktion anwenden (siehe Kapitel 10), um die Daten in einen der ganzzahligen Typen umzuwandeln.

Je nach Vergleichsergebnis werden LT, EQ oder GT auf "wahr" (1) oder "falsch" (0) gesetzt.

Der OK-Ausgang ist durchgeschaltet, wenn die Funktion freigegeben ist und für I1 und I2 eine Zahl eingegeben wurde.



Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für den ersten zu vergleichenden Wert.
I2	I2 enthält eine Konstante oder Referenz für den zweiten zu vergleichenden Wert.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde und I1 und I2 Zahlen sind.
LT	Der Ausgang LT wird durchgeschaltet, wenn I1 kleiner als I2 ist.
EQ	Der Ausgang EQ wird durchgeschaltet, wenn I1 gleich I2 ist.
GT	Der Ausgang GT wird durchgeschaltet, wenn I1 größer als I2 ist.

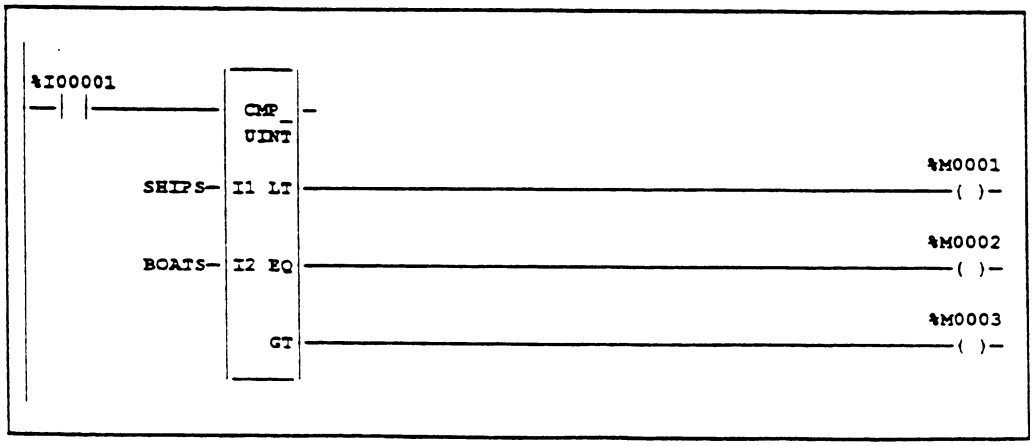
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o		o	•	•	•	•	•	•	
I2	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
LT	•													•
EQ	•													•
GT	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT- oder UINT-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel werden jedesmal, wenn der Eingang %I00001 gesetzt wird, die beiden ganzzahligen Werte SHIPS und BOATS miteinander verglichen. Je nach Vergleichsergebnis werden die internen Merker %M0001, %M0002 oder %M0003 gesetzt.



KAPITEL 7

Bitoperationsfunktionen vergleichen und übertragen Bitfolgen mit einer Minimallänge von einem Wort und einer Maximallänge von 256 Worten oder Doppelworten. Sie benötigen wort- oder doppelwort-strukturierte Daten (WORD oder DWORD). Die Voreinstellung ist WORD.

Obwohl die Daten im 16- oder 32-Bit-Raster angegeben werden müssen, bearbeiten diese Funktionen die Daten als fortlaufendes Bitmuster, wobei Bit 1 des ersten Wortes das niedrigstwertige Bit (LSB) darstellt. Das letzte Bit des letzten Wortes ist das höchstwertige Bit (MSB). Geben Sie zum Beispiel drei Datenworte ab der Referenz %L00100 an, dann werden diese als 48 zusammenhängende Bits behandelt.

%L00100	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 ← bit 1 (LSB)
%L00101	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
%L00102	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
	↑ (MSB)															

HINWEIS

Überlappende Adreßbereiche bei Ein- und Ausgangsreferenzen in Mehrwortfunktionen können unvorhersehbare Ergebnisse verursachen.

In diesem Kapitel werden die folgenden Bitoperationsfunktionen beschrieben:

Abkürzung	Funktion
AND	Logisch UND
OR	Logisch ODER
XOR	Logische Antivalenz
NOT	Logische Invertierung
SHL	Nach links verschieben
SHR	Nach rechts verschieben
ROL	Nach links rotieren
ROR	Nach rechts rotieren
BTST	Bit testen
BSET	Ein Bit auf 1 setzen
BCLR	Ein Bit auf 0 setzen
BPOS	Ein auf 1 gesetztes Bit finden
MKCOMP	Maskierter Vergleich

AND und OR (WORD, DWORD)

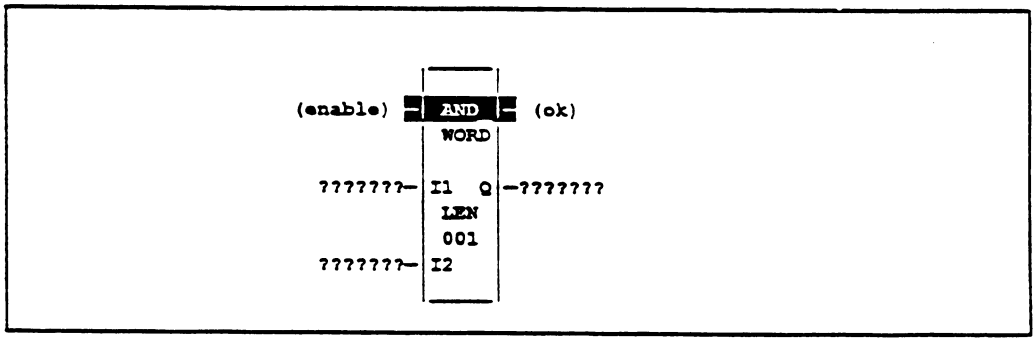
Beginnend mit dem jeweils ersten Bit (niedrigste Adresse) untersuchen die Funktionen AND und OR bei jedem Zyklus, bei dem sie Stromfluß erhalten, die einzelnen Bits in den Bitfolgen I1 und I2.

Sind zwei einander entsprechende Bits beide "1", dann trägt die AND-Funktion an der zugehörigen Stelle in der Ausgangsbitfolge Q eine "1" ein. Ist eines der beiden Bits "0", dann wird an der zugehörigen Stelle in der Ausgangsbitfolge Q eine "0" eingetragen. Die AND-Funktion ist bei der Erstellung von Masken oder Menüs hilfreich, bei denen nur bestimmte Bits durchgelassen werden (die einer "1" in der Maske entsprechen) und alle anderen Bits auf "0" gesetzt werden. Mit dieser Funktion kann auch ein bestimmter Bereich im Wortspeicher gelöscht werden, indem die Bits mit einer anderen Bitfolge, die nur Nullen enthält, über die AND-Funktion verknüpft werden. Die angegebenen Bitfolgen können sich überlappen.

Ist bei der OR-Funktion eines der beiden Bits "1", dann wird an der zugehörigen Stelle in der Ausgangsbitfolge Q eine "1" eingetragen. Sind beide Bits "0", dann wird in der Ausgangsbitfolge Q an der entsprechenden Stelle eine "0" eingetragen. Die OR-Funktion ist hilfreich bei der Kombination von Bitfolgen und bei der Steuerung mehrerer Ausgänge durch eine einfache logische Struktur. Die Funktion entspricht einer Parallelschaltung zweier Relaiskontakte für jedes in der Bitfolge enthaltene Bit. Mit ihr können Anzeigelampen aus Eingangszuständen heraus angesteuert oder Statuslampen zum Blinken gebracht werden.

Bei beiden Funktionen kann die Länge der Zeichenfolge bis zu 256 Worte oder Doppelworte betragen.

Die Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für das erste Wort.
I2	I2 enthält eine Konstante oder Referenz für das zweite Wort.
OK	Der OK-Ausgang wird jedesmal durchgeschaltet, wenn das Freigabesignal (enable) aktiv ist.
Q	Der Ausgang Q enthält das Ergebnis der logischen Verknüpfung von I1 und I2.

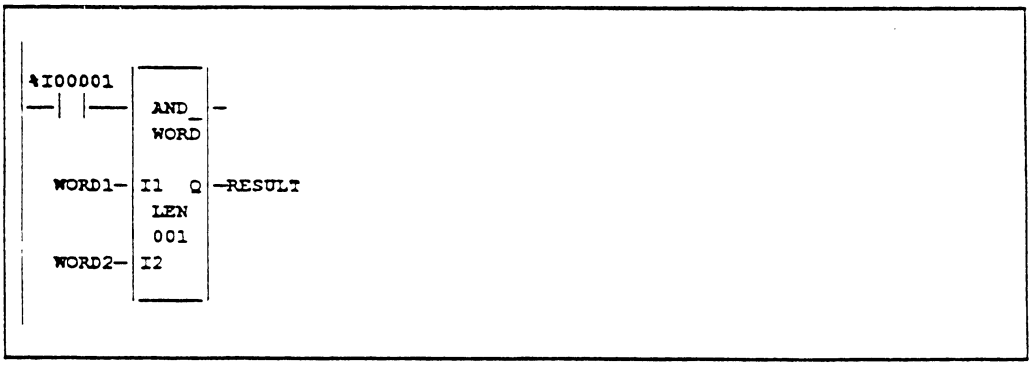
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o	o	o	•	•	•	•	•	•	
I2	•	o	o	o	o	o	o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o	o†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Bei dem folgenden Beispiel werden jedesmal, wenn der Eingang %I00001 gesetzt wird, die durch die symbolischen Adressen WORD1 und WORD2 spezifizierten Bitfolgen (je 16 Bits) untersucht. Das Ergebnis der logischen AND-Verknüpfung wird in der Ausgangs-Bitfolge RESULT eingetragen.



WORD1	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
WORD2	1	1	0	1	1	1	0	0	0	0	0	0	1	1	1	1
RESULT	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0

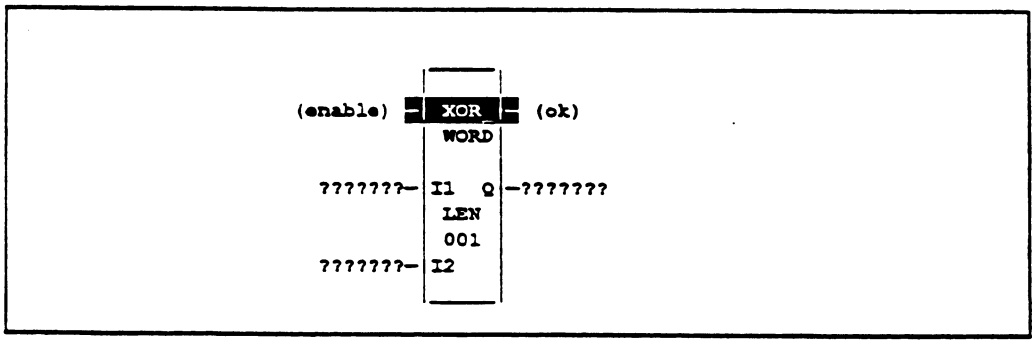
XOR (WORD, DWORD)

Die XOR-Funktion vergleicht die einzelnen einander entsprechenden Bits in den Bitfolgen I1 und I2 miteinander. Unterscheiden sich diese beiden Bits, dann wird an der zugehörigen Stelle in der Ausgangsbitfolge Q eine "1" eingetragen.

Beginnend mit dem jeweils ersten Bit (niedrigste Adresse) untersucht die XOR-Funktion bei jedem Zyklus, bei dem sie Stromfluß erhält, die einzelnen einander entsprechenden Bits in den Bitfolgen I1 und I2. Ist nur eines der beiden Bits "1", dann wird an der zugehörigen Stelle in der Ausgangsbitfolge Q eine "1" eingetragen. Ein String kann bis zu 256 Worte oder Doppelworte umfassen. Die XOR-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.

Beginnen Bitfolge I2 und Ausgangsbitfolge Q mit der gleichen Referenz, dann wird bei einer "1" in der Bitfolge I1 bei jedem Zyklus, bei dem die Funktion Stromfluß erhält, das entsprechende Bit in der Bitfolge I2 zwischen "0" und "1" umgeschaltet. Es können längere Zyklen programmiert werden, indem der Stromfluß zur Funktion mit der doppelten gewünschten Blinkfrequenz gepulst wird. Der Stromflußimpuls sollte mindestens einen Zyklus lang sein (Wischrelais oder automatisch rücksetzendes Zeitglied).

Mit der XOR-Funktion kann ein schneller Vergleich zweier Bitfolgen durchgeführt werden oder es kann eine Bitgruppe mit einer Frequenz von einem EIN-Zustand pro zwei Zyklen zum Blinken angeregt werden.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für das erste Wort, das über die XOR-Funktion verknüpft wird.
I2	I2 enthält eine Konstante oder Referenz für das zweite Wort, das über die XOR-Funktion verknüpft wird.
OK	Der OK-Ausgang wird jedesmal durchgeschaltet, wenn das Freigabesignal (enable) aktiv ist.
Q	Der Ausgang Q enthält das Ergebnis der logischen XOR-Verknüpfung von I1 und I2.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o	o	o	•	•	•	•	•	•	
I2	•	o	o	o	o	o	o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o	o†	o	•	•	•		•	•	

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

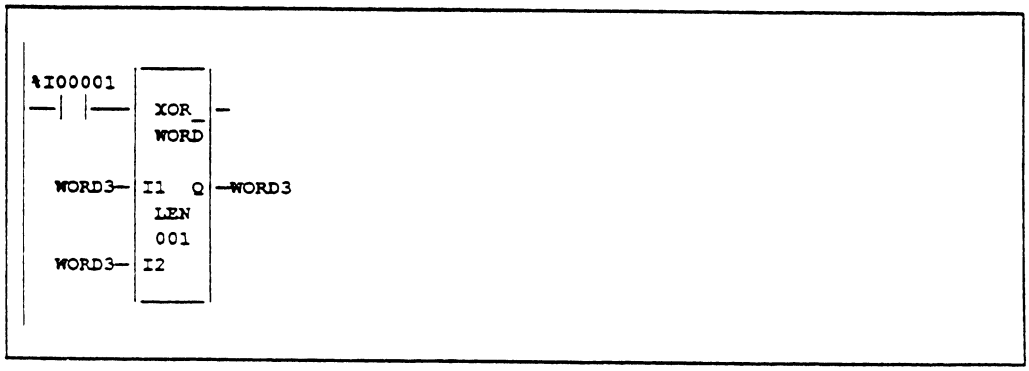
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, die durch die symbolische Adresse WORD3 dargestellte Bitfolge gelöscht (auf Null gesetzt).



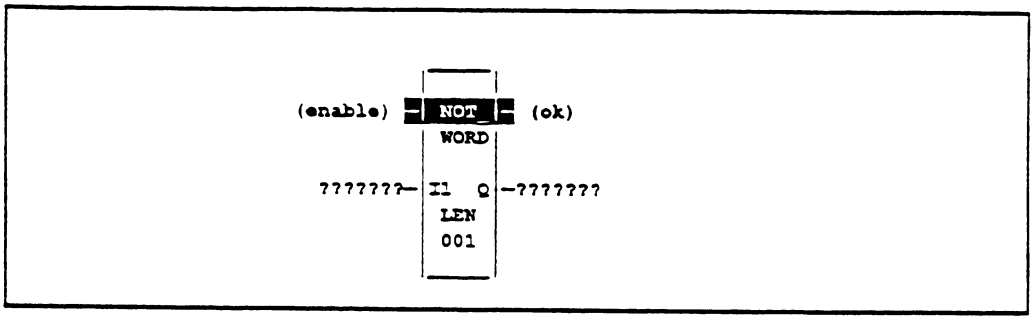
I1 (WORD3)	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
I2 (WORD3)	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
Q (WORD3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NOT (WORD, DWORD)

Die NOT-Funktion trägt in die einzelnen Stellen der Ausgangs-Bitfolge Q den invertierten Wert der Bits der Eingangsbitfolge I1 ein.

Bei jedem Zyklus, bei dem die Funktion Stromfluß erhält, werden alle Bits verändert, so daß die Ausgangs-Bitfolge Q ein gespiegeltes Abbild der Bitfolge I1 ist. Ein String kann bis zu 256 Worte oder Doppelworte umfassen.

Die NOT-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die Operation wird durchgeführt, wenn die Funktion freigegeben ist.
I1	I1 enthält eine Konstante oder Referenz für das Wort, das invertiert werden soll.
OK	Der OK-Ausgang wird jedesmal durchgeschaltet, wenn das Freigabesignal (enable) aktiv ist.
Q	Der Ausgang Q enthält die Negation von I1.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o	o	o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o	o†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

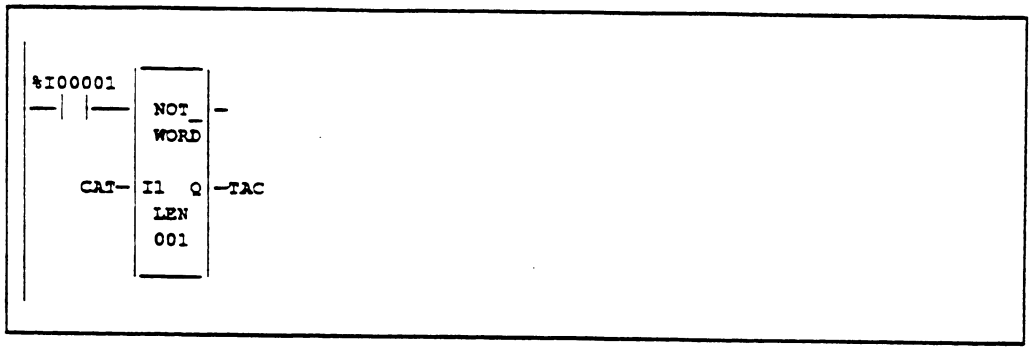
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

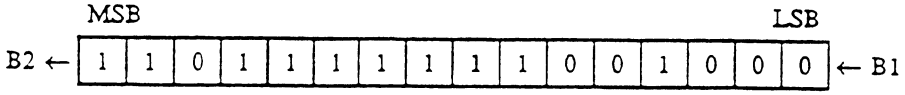
Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, die durch die symbolische Adresse TAC dargestellte Bitfolge in die invertierte Bitfolge CAT umgewandelt.

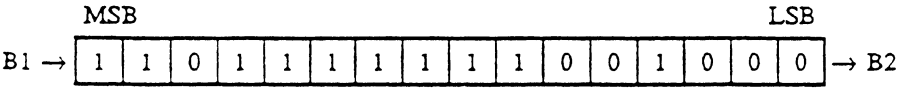


SHL und SHR (WORD, DWORD)

Die SHL-Funktion verschiebt alle Bits eines Wortes oder einer Gruppe um eine vorgegebene Anzahl Stellen nach links. Beim Verschieben wird die angegebene Anzahl Bits aus dem Ausgangsstring nach links herausgeschoben. Die gleiche Anzahl Bits, die oben herausgeschoben wird, wird am unteren Ende wieder aufgefüllt.



Die SHR-Funktion verschiebt alle Bits eines Wortes oder einer Gruppe um eine vorgegebene Anzahl Stellen nach rechts. Beim Verschieben wird die angegebene Anzahl Bits aus dem Ausgangsstring nach rechts herausgeschoben. Die gleiche Anzahl Bits, die unten herausgeschoben wird, wird am oberen Ende wieder aufgefüllt.



Für beide Funktionen kann eine Stringlänge zwischen 1 und 256 Worten oder Doppelworten eingestellt werden.

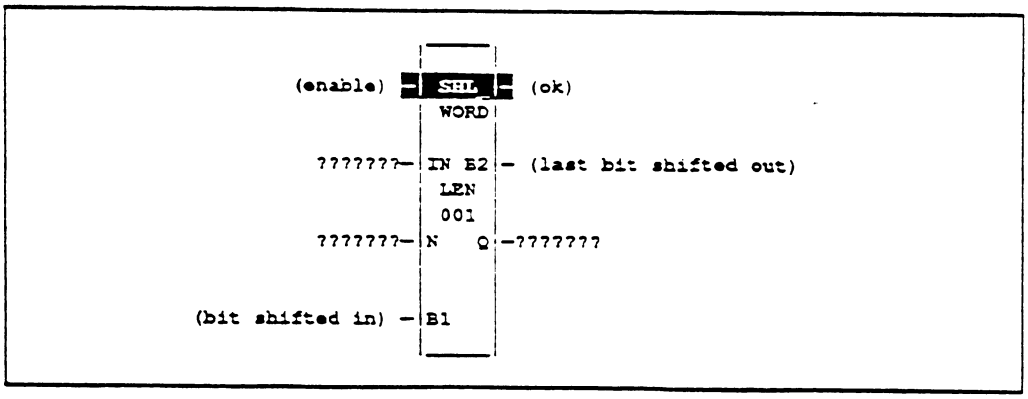
Die Anzahl der Plätze, die für die Verschiebung angegeben wird, muß größer als 0 und kleiner als die Anzahl Bits in der Bitfolge sein. Ist dies nicht der Fall, erfolgt keine Verschiebung und es findet kein Stromfluß statt.

Die in den Stringanfang eingeschobenen Bits werden über den Eingabeparameter B1 festgelegt. Ist die für die Verschiebung angegebene Anzahl Stellen größer als 1, dann haben alle eingeschobenen Bits den gleichen Wert (0 oder 1). Es sind hier möglich;

- Der Boolsche Ausgangswert einer anderen Programmfunktion.
- Nur Einsen. Hierzu müssen Sie die Spezialreferenz mit der symbolischen Adresse ALW-ON als Eingabe zu Eingang B1 angeben.
- Nur Nullen. Hierzu müssen Sie die Spezialreferenz mit der symbolischen Adresse ALW-OFF als Eingabe zu Eingang B1 angeben.

SHL und SHR schalten den Stromfluß nach rechts durch, solange die angegebene Anzahl verschobener Bits nicht größer als die Gesamtlänge des Strings und von Null verschieden ist.

Der Funktionsausgang Q ist die verschobene Kopie des Eingabestrings. Ausgangswert B2 ist das letzte hinausgeschobene Bit. Wurden z.B. vier Bits verschoben, dann ist B2 das vierte hinausgeschobene Bit.



Parameter:

Parameter	Beschreibung
enable	Die Verschiebung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält das erste Wort, das verschoben werden soll.
N	N enthält die Anzahl Plätze, um die das Feld verschoben werden soll.
B1	B1 enthält den Bitwert, der in das Feld eingeschoben werden soll.
B2	B2 enthält den Wert des letzten Bits, das aus dem Feld herausgeschoben werden soll.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Verschiebung aktiviert wurde und die Verschiebelänge nicht größer als die Feldgröße ist.
Q	Der Ausgang Q enthält das erste Wort des verschobenen Feldes.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o	o	o	•	•	•	•	•	•	
N	•	•	•	•	•		•	•	•	•	•	•	•	
B1	•													•
B2	•													•
ok	•													•
Q	•	o	o	o	o	o†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

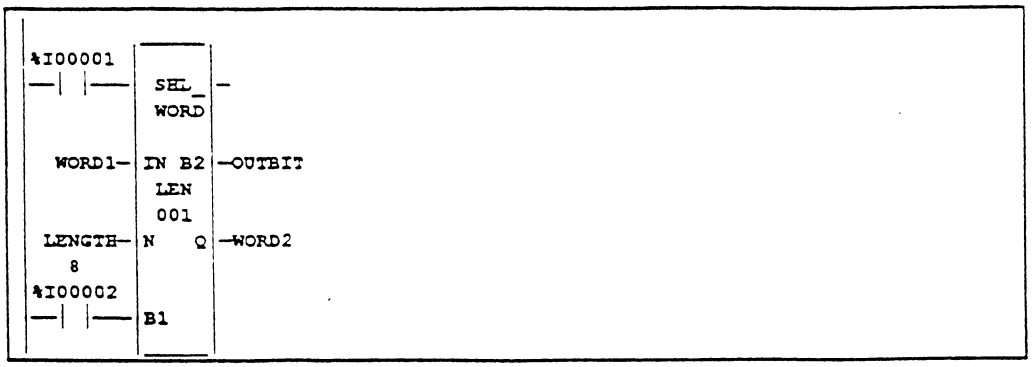
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, die durch die symbolische Adresse WORD2 dargestellte Ausgangsbitfolge als Kopie der um die durch die symbolische Adresse LENGTH angegebene Anzahl Stellen nach links verschobenen Bitfolge WORD1 erstellt. Die resultierenden leeren Bitstellen werden mit Nullen aufgefüllt.



ROL und ROR (WORD, DWORD)

Mit der ROL-Funktion können alle Bits in einer Bitfolge um eine vorgegebene Anzahl Stellen nach links rotiert werden. Bei der Rotation wird die angegebene Anzahl Bits nach links aus dem Ausgangsstring heraus und auf der rechten Seite wieder in den String hinein geschoben.

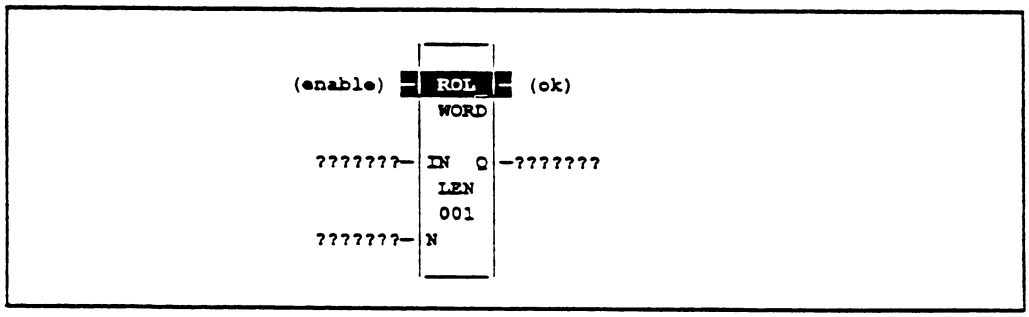
Mit der ROR-Funktion werden die Bits im String nach rechts rotiert. Bei der Rotation wird die angegebene Anzahl Bits nach links aus dem Ausgangsstring heraus und auf der rechten Seite wieder in den String hinein geschoben.

Bei beiden Funktionen kann eine Stringlänge zwischen 1 und 256 Worten oder Doppelworten eingestellt werden.

Die für die Rotation angegebene Anzahl Stellen muß größer als 0 und kleiner als die Anzahl der im String enthaltenen Bits sein. Ist dies nicht der Fall, erfolgt keine Rotation und es findet kein Stromfluß statt.

Die Funktionen ROL und ROR schalten den Stromfluß nach rechts durch, solange die angegebene Anzahl rotierter Bits nicht größer als die Gesamtlänge des Strings und nicht kleiner als Null ist.

Das Ergebnis wird in den Ausgangsstring Q eingetragen, der ursprüngliche String wird dabei nicht verändert. Der gesamte String wird bei jedem Zyklus, bei dem die Funktion Stromfluß erhält, rotiert.



Parameter:

Parameter	Beschreibung
enable	Rotation wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält das erste Wort, das rotiert werden soll.
N	N enthält die Anzahl Plätze, um die das Feld rotiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Rotation aktiviert ist und die Rotationslänge nicht größer als die Feldlänge ist.
Q	Ausgang Q enthält das erste Wort des rotierten Feldes.

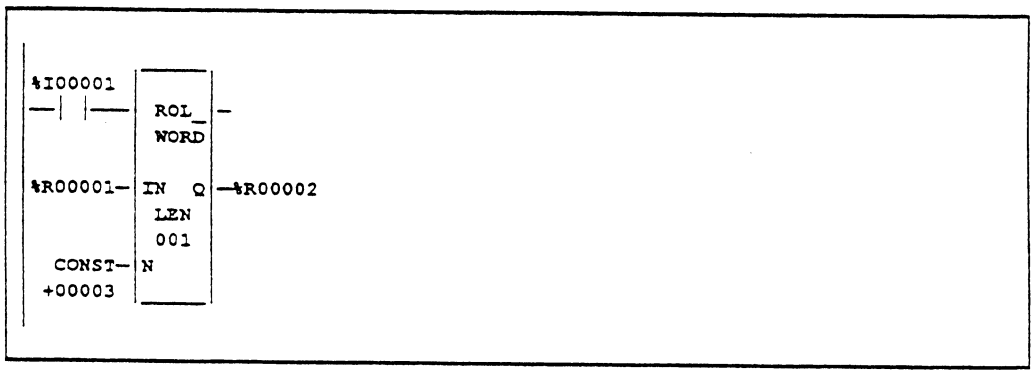
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o	o	o	•	•	•	•	•	•	
N	•	•	•	•	•		•	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o	o†	o	•	•	•	•	•		

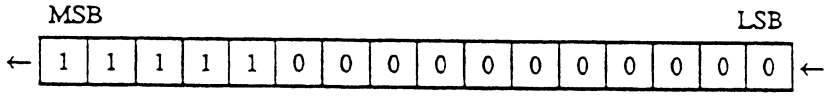
Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

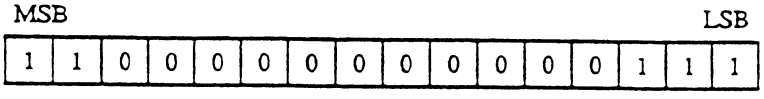
Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, die Eingangsbitfolge %R00001 um 3 Bits rotierend verschoben. Das Ergebnis wird in %R00002 abgelegt. Nachdem die Funktion ausgeführt wurde, bleibt die Eingangsbitfolge %R00001 unverändert. Wird die gleiche Referenz für IN und Q verwendet, dann findet eine Rotation auf der Stelle statt.



%R00001:



%R00002 (nachdem %I00001 gesetzt wurde):

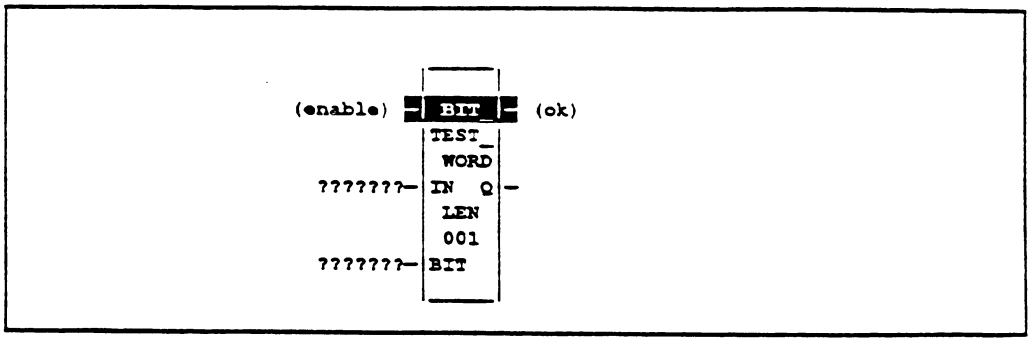


BTST (WORD, DWORD)

Mit der BTST-Funktion können Sie feststellen, ob ein Bit in einem String momentan 0 oder 1 ist. Das Ergebnis des Tests wird in Ausgang Q eingetragen.

Bei jedem Zyklus, bei dem die Bittestfunktion Stromfluß empfängt, setzt sie ihren Ausgang auf den gleichen Zustand, den das angegebene Bit innehat. Wird zur Angabe der Bitnummer statt einer Konstanten ein Register verwendet, dann kann die gleiche Programmfunktion in verschiedenen Zyklen unterschiedliche Bits testen. Q wird auf "falsch" gesetzt, wenn der Wert von BIT außerhalb des Bereichs (1 ≤ BIT ≤ (16*LEN)) liegt.

Es kann eine Stringlänge zwischen 1 und 256 Worten oder Doppelworten eingestellt werden.



Parameter

Parameter	Beschreibung
enable	Der Bittest wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält das erste Datenwort für die Funktion.
BIT	BIT enthält die Nummer des Bits aus IN, das getestet werden soll. Zulässige Werte sind (1 ≤ BIT ≤ (16*LEN)).
OK	Der OK-Ausgang wird durchgeschaltet, wenn der Bittest aktiviert und LEN größer als Null ist.
Q	Der Ausgang Q wird durchgeschaltet, wenn des getestete Bit "1" war.

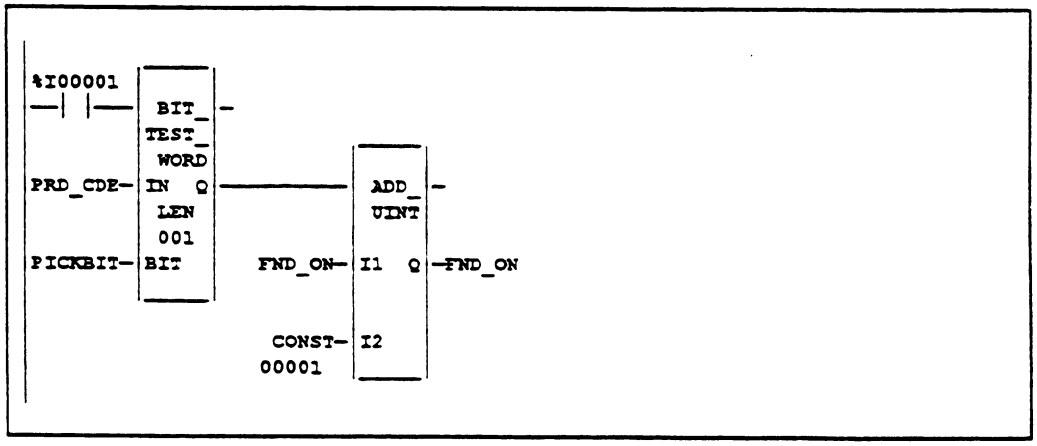
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o	o	o	•	•	•	•	•	•	
BIT	•	•	•	•	•		•	•	•	•	•	•	•	
ok	•													•
Q	•													

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, das Bit an der in Referenz PICKBIT enthaltenen Adresse getestet. Ist dieses Bit, das Teil des Strings PRD_CDE ist, "1", dann wird der Ausgang Q durchgeschaltet und der Stromfluß wird zur ADD-Funktion weitergeleitet. Der aktuelle Wert des Eingangs I1 der ADD-Funktion wird dann um 1 erhöht.

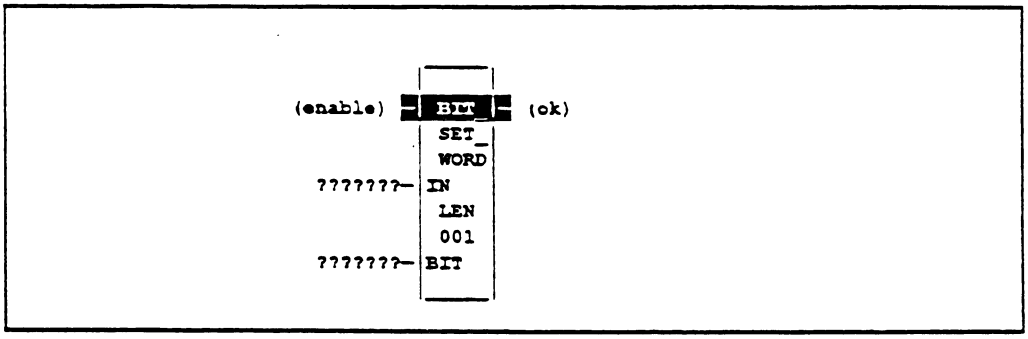


BSET und BCLR (WORD, DWORD)

Mit der BSET-Funktion können Sie ein Bit in einem String auf "1" setzen. Mit der BCLR-Funktion können Sie ein Bit in einem String auf "0" setzen.

Bei jedem Zyklus, bei dem Stromfluß durchgeschaltet wird, setzt die BSET-Funktion das angegebene Bit auf "1" und die BCLR-Funktion das angegebene Bit auf "0". Wird zur Angabe der Bitnummer keine Konstante, sondern eine Variable (Register) verwendet, dann kann die gleiche Programmfunktion in verschiedenen Zyklen unterschiedliche Bits auf "1" bzw. "0" setzen.

Es kann eine Stringlänge zwischen 1 und 256 Worten oder Doppelworten eingestellt werden. Die Funktion schaltet den Stromfluß weiter nach rechts, wenn der Wert von BIT innerhalb des Bereichs ($1 \leq BIT \leq (16*LEN)$) liegt. Ist dies nicht der Fall, dann wird OK auf "falsch" gesetzt.



Parameter:

Parameter	Beschreibung
enable	Ein Bit wird auf 1 gesetzt, wenn die Funktion freigegeben ist.
IN	IN enthält das erste Datenwort für die Funktion.
BIT	BIT enthält die Nummer des Bits aus IN, das auf 1 gesetzt werden soll. Zulässige Werte sind ($1 \leq BIT \leq (16*LEN)$).
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion aktiviert ist.

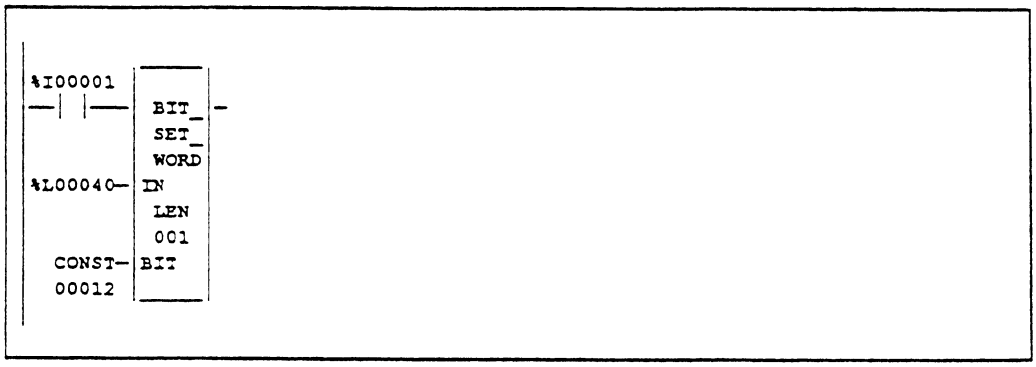
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN		o	o	o	o	o†	o	•	•	•	•	•		
BIT	•	•	•	•	•		•	•	•	•	•	•	•	
ok	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.

Beispiel:

Bei dem folgenden Beispiel wird jedesmal, wenn der Eingang %I00001 gesetzt wird, Bit 12 des bei Referenz %L00040 beginnenden Strings auf "1" gesetzt.



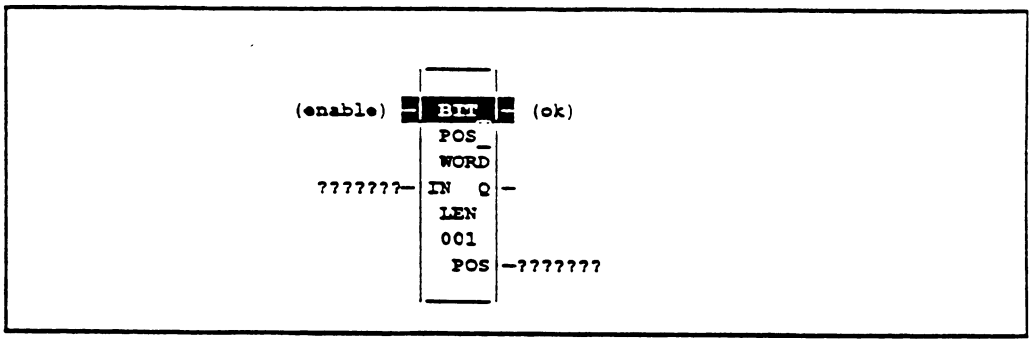
BPOS (WORD, DWORD)

Mit der BPOS-Funktion kann ein auf "1" gesetztes Bit innerhalb einer Bitfolge lokalisiert werden.

Bei jedem Zyklus, bei dem die Funktion Stromfluß empfängt, sucht sie den bei IN beginnenden String ab. Hält die Funktion mit der Suche inne, dann hat sie entweder ein auf "1" gesetztes Bit gefunden oder das Ende des Strings erreicht.

POS wird auf die Position in der Bitfolge gesetzt, in der das erste von Null verschiedene Bit angetroffen wurde. POS wird auf Null gesetzt, wenn kein von Null verschiedenes Bit gefunden wurde.

Es kann eine Stringlänge zwischen 1 und 256 Zeichen eingestellt werden. Die Funktion schaltet den Stromfluß nach rechts durch, wenn der Freigabeeingang "wahr" ist.



Parameter	Beschreibung
enable	Die Bitsuche wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält das erste zu bearbeitende Datenwort.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion aktiviert ist.
POS	Die Position des ersten von Null verschiedenen Bits, POS ist Null, wenn kein von Null verschiedenes Bit gefunden wurde.
Q	Der Ausgang Q wird durchgeschaltet, wenn ein auf "1" gesetztes Bit gefunden wurde.

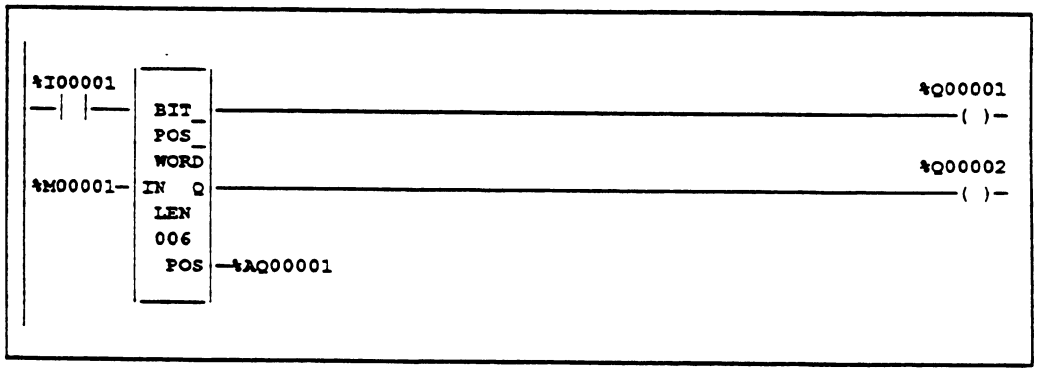
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o	o	o	•	•	•	•	•	•	
POS	•	•	•	•	•		•	•	•	•	•	•		
ok	•													•
Q	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.

Beispiel:

Wird bei dem folgenden Beispiel %I00001 gesetzt, dann wird die bei %M00001 beginnende Bitfolge abgesucht, bis ein auf "1" gesetztes Bit gefunden wurde oder bis 6 Worte abgesucht wurden. Spule %Q00001 wird durchgeschaltet. Wird ein auf "1" gesetztes Bit gefunden, dann wird seine Lage innerhalb der Bitfolge in %AQ00001 eingetragen und %Q00002 wird durchgeschaltet. Ist %I00001 gesetzt, Bit %M00001 "0" und Bit %M00002 "1", dann ist der in %AQ00001 eingetragene Wert 2.



MCMP (WORD, DWORD)

Mit der MCMP-Funktion kann der Inhalt zweier Strings miteinander verglichen werden. In Eingangsstring I1 können z.B. die Zustände von Ausgängen (Spulen oder Anlasser) enthalten sein, während Eingangsstring I2 die Rückkopplung der Eingangszustände (Endschalter oder Kontakte) enthält.

Bei jedem Zyklus, bei dem Stromfluß zur Funktion durchgeschaltet wird, vergleicht diese Funktion die Bits im ersten String mit den entsprechenden Bits im zweiten String. Der Vergleich wird solange fortgesetzt, bis eine Diskrepanz erkannt wird oder bis das Stringende erreicht ist.

Über den BIT-Eingang wird die Bitnummer gespeichert, bei der der nächste Vergleich beginnen soll. Da die Bitnummer der letzten Diskrepanz im Ausgang BN gespeichert wird, kann die gleiche Referenz für BIT und BN verwendet werden.

Wollen Sie den nächsten Vergleich an einer anderen Stelle beginnen, dann können Sie für BIT und BN unterschiedliche Referenzen verwenden. Wird bei BIT eine Adresse außerhalb des Strings angegeben, dann wird BIT vor dem nächsten Vergleich auf 1 rückgesetzt.

Die Funktion schaltet den Stromfluß immer nach rechts durch, wenn sie Stromfluß empfängt. Die übrigen Ausgänge der Funktion hängen entsprechend nachstehender Beschreibung von den Zuständen der entsprechenden Maskenbits ab.

Alle Bits in I1 und I2 sind gleich

Stimmen alle einander entsprechenden Bits in I1 und I2 überein, dann setzt die Funktion den Diskrepanzausgang MC auf 0 und BN auf die höchste Bitnummer der Eingabestrings. Der Vergleich ist dann beendet. Beim nächsten Aufruf der MCMP-Funktion wird der Ausgang auf 1 rückgesetzt.

Es wurde eine Diskrepanz festgestellt

Sind zwei miteinander verglichene Bits ungleich, dann überprüft die Funktion das entsprechend numerierte Bit im String M (der Maske). Ist das Maskenbit "1", dann wird der Vergleich fortgesetzt, bis eine weitere Diskrepanz gefunden oder das Stringende erreicht wurde.

Wird eine Diskrepanz festgestellt und ist das entsprechende Maskenbit eine "0", dann

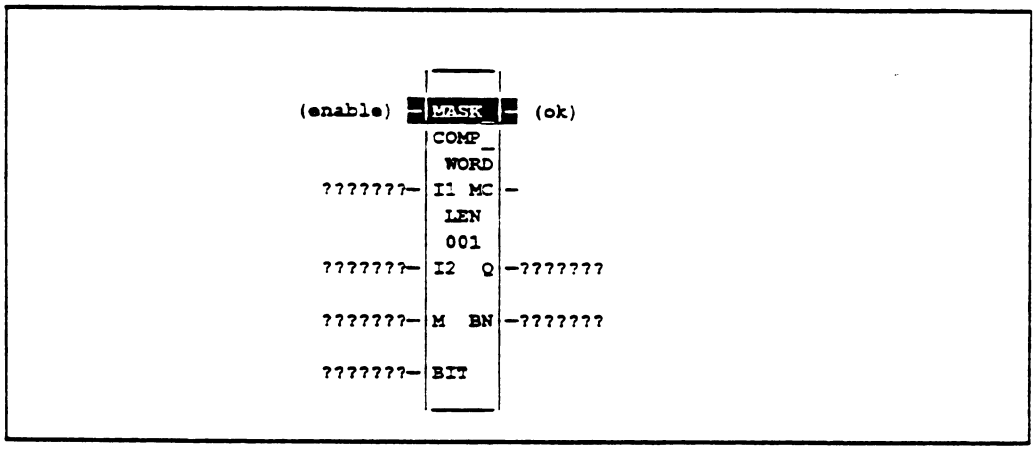
1. Setzt die Funktion das entsprechende Maskenbit auf "1".

2. Setzt die Funktion den Diskrepanzausgang (MC) auf "1".

3. Aktualisiert die Funktion den Ausgangs-Bitstring Q so, daß er dem neuen Inhalt des Maskenstrings M entspricht.

4. Setzt die Funktion den Bitnummernausgang (BN) auf die Bitnummer, bei der die Diskrepanz auftrat (z.B. 6).

5. Bricht die Funktion den Vergleich ab.



Parameter:

Parameter	Beschreibung
enable	Freigabelogik der Funktion.
I1	Referenz des ersten Bitstrings im Vergleich.
I2	Referenz des zweiten Bitstrings im Vergleich.
M	Referenz der Bitstringmaske.
BIT	Referenz der Bitnummer, bei der der nächste Vergleich beginnen soll.
ok	Der ok-Ausgang wird durchgeschaltet, wenn enable aktiviert ist.
MC	Anwenderlogik, mit der eine Diskrepanz festgestellt wird.
Q	Ausgangskopie des Masken-Bitstrings (M).
BN	Nummer des Bits, bei dem die letzte Diskrepanz auftrat.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
I1	•	o	o	o	o	o	o	•	•	•	•	•		
I2	•	o	o	o	o	o	o	•	•	•	•	•		
M		o	o	o	o	o†	o	•	•	•	•	•		
BIT	•	•	•	•	•		•	•	•	•	•	•	•	
ok	•													•
MC	•													•
Q		o	o	o	o	o†	o	•	•	•	•	•		
BN		•	•	•	•		•	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

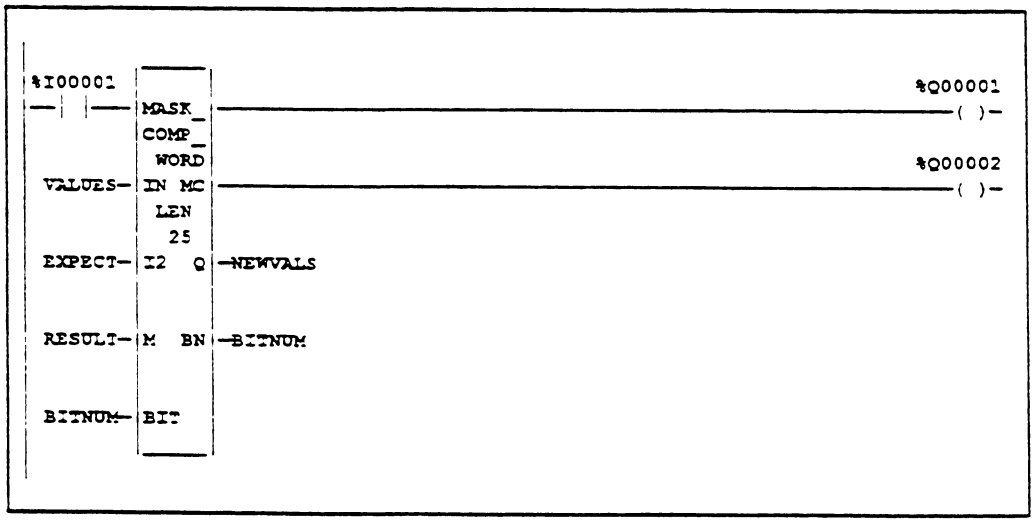
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Bei dem folgenden Beispiel werden jedesmal, wenn der Eingang %I00001 gesetzt wird, die von der Referenz VALUES dargestellten Bits mit den von der Referenz EXPECT dargestellten Bits verglichen. Der Vergleich beginnt mit dem in BITNUM angegebenen Bit. Wird eine nicht maskierte Diskrepanz erkannt, dann stoppt der Vergleich und das entsprechende Bit wird in der Maske RESULT gesetzt. Zusätzlich wird der Ausgabestring NEWVALS mit dem neuen Wert von RESULT aktualisiert und die Spule %Q00002 wird durchgeschaltet. Spule %Q00001 wird immer dann durchgeschaltet, wenn die Funktion Stromfluß empfängt.



KAPITEL 8

Die Datenverschiebungsfunktionen vermitteln die grundlegenden Möglichkeiten zur Bewegung von Daten. In diesem Kapitel werden folgende Datenverschiebungsfunktionen beschrieben:

Abkürzung	Funktion	Beschreibung
MOVE	Kopieren	Diese Funktion kopiert Daten bitweise. Die maximal zulässige Länge beträgt 32.767. Daten können ohne vorherige Konvertierung in einen anderen Datentyp kopiert werden.
BLKMOV	Block verschieben	Diese Funktion kopiert einen Block von sieben Konstanten in eine angegebene Speicheradresse. Die Konstanten werden als Teil der Funktion eingegeben.
BLKCLR	Block löschen	Diese Funktion überschreibt den Inhalt eines Datenblocks mit Nullen. Sie erlaubt das Löschen eines bit- (%I, %Q, %M, %G oder %T) oder wortstrukturierten (%R, %P, %L, %AI oder %AQ) Speicherbereichs. Die maximal zulässige Länge beträgt 256.
SHFREG	Schieberegister	Diese Funktion verschiebt ein oder mehrere Datenworte in eine Tabelle. Die maximal zulässige Länge beträgt 256.
BITSEQ	Bitfolgesteuerung	Diese Funktion verschiebt eine Bitfolge durch ein Bitfeld. Die maximal zulässige Länge beträgt 256.
COMMREQ	Kommunikations-anforderung	Diese Funktion ermöglicht dem Programm den Datenaustausch mit einem intelligenten Modul, wie zum Beispiel einem Buscontroller, programmierbaren Coprozessormodul oder Subnet-Modul.
VMERD	VME lesen	Liest Daten von der VME-Rückwandplatine. Die maximal zulässige Länge beträgt 32.767.
VMEWRT	VME schreiben	Schreibt Daten zur VME-Rückwandplatine. Die maximal zulässige Länge beträgt 32.767.
VMERMW	VME lesen/ändern/ schreiben	Aktualisieren eines Datenelements mit dem Lesen/Andern/Schreiben-Zyklus am VME-Bus.
VMETS	VME testen und setzen	Bearbeitung von Semaphoren am VME-Bus.
SWAP	Vertauschen	Diese Funktion vertauscht die beiden Bytes eines Wortes oder die beiden Worte eines Doppelworts. Die maximal zulässige Länge beträgt 256.

Zuordnung von Referenzadressen

Überlappende Ein- und Ausgangsadressen bei Mehrwortfunktionen können zu unvorhersehbaren Ergebnissen führen.

MOVE (INT, DINT, UINT, BIT, WORD, DWORD, REAL)

Mit der MOVE-Funktion können Sie Daten bitweise von einer Stelle zur andern kopieren. Da die Daten im Bitformat kopiert werden, können Quelle und Ziel unterschiedliche Datentypen besitzen.

Die MOVE-Funktion besitzt zwei Ein- und zwei Ausgangsparameter. Erhält die Funktion Stromfluß, dann kopiert sie Daten vom Eingangsparameter IN bitweise zum Ausgangsparameter Q.

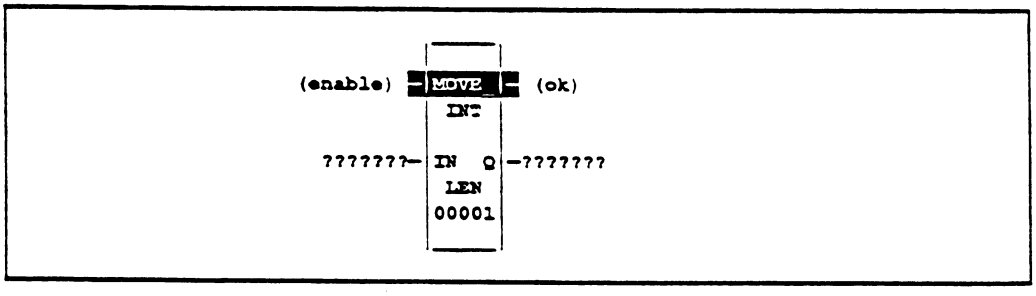
Werden Daten von einer Stelle im diskreten Speicher zu einer anderen Stelle kopiert (z.B. vom %I-Speicher zum %T-Speicher), dann wird die mit den diskreten Speicherelementen verbundene Transitionsinformation ebenfalls zur neuen Stelle kopiert. Die Daten im Eingangsparameter werden nur bei einer Adreßüberlappung im Ziel verändert.

Der IN-Eingang der Funktion kann entweder eine Referenz für die zu kopierenden Daten oder eine Konstante sein. Wird eine Konstante angegeben, dann wird der gleiche konstante Wert in der durch die Ausgangsreferenz angegebenen Speicheradresse eingetragen. Wurde zum Beispiel für IN der Wert 4 angegeben, dann wird diese 4 in der durch die Ausgangsreferenz Q angegebenen Speicheradresse eingetragen. Wird bei einer Länge > 1 eine Konstante angegeben, dann wird diese Konstante ab der durch Ausgang Q angegebenen Speicheradresse über die angegebene Länge hinweg eingetragen. Wurde zum Beispiel für IN der Wert 9 und eine Länge von 4 angegeben, dann wird diese 9 in der durch Ausgang Q spezifizierten Speicheradresse und den darauf folgenden drei Speicheradressen eingetragen.

Bei den Funktionen MOVE_INT, MOVE_DINT, MOVE_UINT, MOVE_WORD, MOVE_DWORD und MOVE_REAL muß der Wert für LEN zwischen 1 und 32.767 liegen. Der Operand LEN gibt an, wieviele Worte verschoben werden sollen.

Wird bei MOVE_BIT der Parameter IN als Konstante angegeben, dann muß der Wert von LEN zwischen 1 und 16 liegen. Der Operand LEN gibt hier an, wieviele Bits verschoben werden sollen.

Die MOVE-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die Verschiebung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält den Wert, der verschoben werden soll. Bei MOVE_BIT kann jede diskrete Referenz verwendet werden, die nicht im Byteraster zu liegen braucht. Es werden jedoch ab der angegebenen Referenzadresse 16 Bits on-line angezeigt.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist.
Q	Der an IN anstehende Wert wird in Q eingetragen wenn die Funktion ausgeführt wird. Bei MOVE_BIT kann jede diskrete Referenz verwendet werden, die nicht im Byteraster zu liegen braucht. Es werden jedoch ab der angegebenen Referenzadresse 16 Bits on-line angezeigt.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o	Δ	o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o	†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Datentyp nur für INT, UINT oder WORD-Daten zulässig.
Bei MOVE_BIT brauchen die diskreten Speichertypen %I, %Q, %M und %T nicht im Byteraster zu liegen.

Δ = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

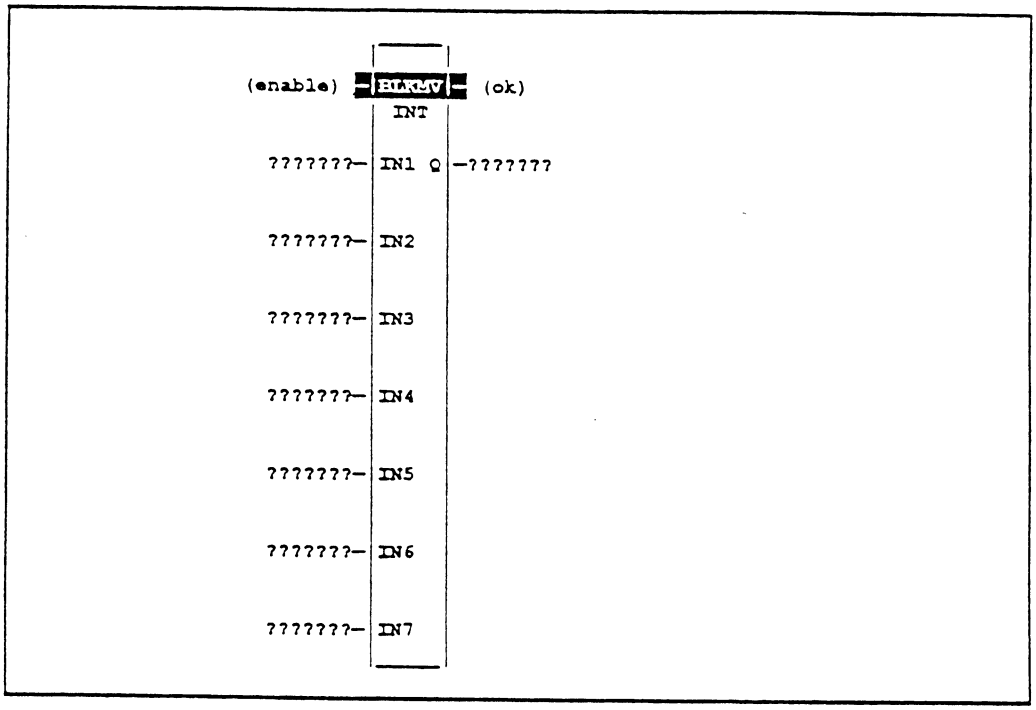
Wird in diesem Beispiel %I00003 gesetzt, dann werden die drei Bits %M00001, %M00002 und %M00003 zu %M00100, %M00101 und %M00102 kopiert und die Spule %Q00001 wird durchgeschaltet.

BLKMOV (INT, DINT, UINT, WORD, DWORD, REAL)

Mit der BLKMOV-Funktion können Sie einen Block von sieben Konstanten zu einer angegebenen Speicheradresse kopieren.

Die BLKMOV-Funktion besitzt acht Ein- und zwei Ausgangsparameter. Erhält die Funktion Stromfluß, dann kopiert sie die Konstanten in einen zusammenhängenden Speicherbereich, der bei der im Ausgang Q spezifizierten Adresse beginnt. Ausgang Q kann nicht der Eingang einer anderen Programmfunktion sein.

Die BLKMOV-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die Verschiebung wird durchgeführt, wenn die Funktion freigegeben ist.
IN1-IN7	IN1-IN7 enthalten sieben Konstanten.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist.
Q	Ausgang Q enthält die erste ganze Zahl des kopierten Feldes. IN1 wird zu Q kopiert.

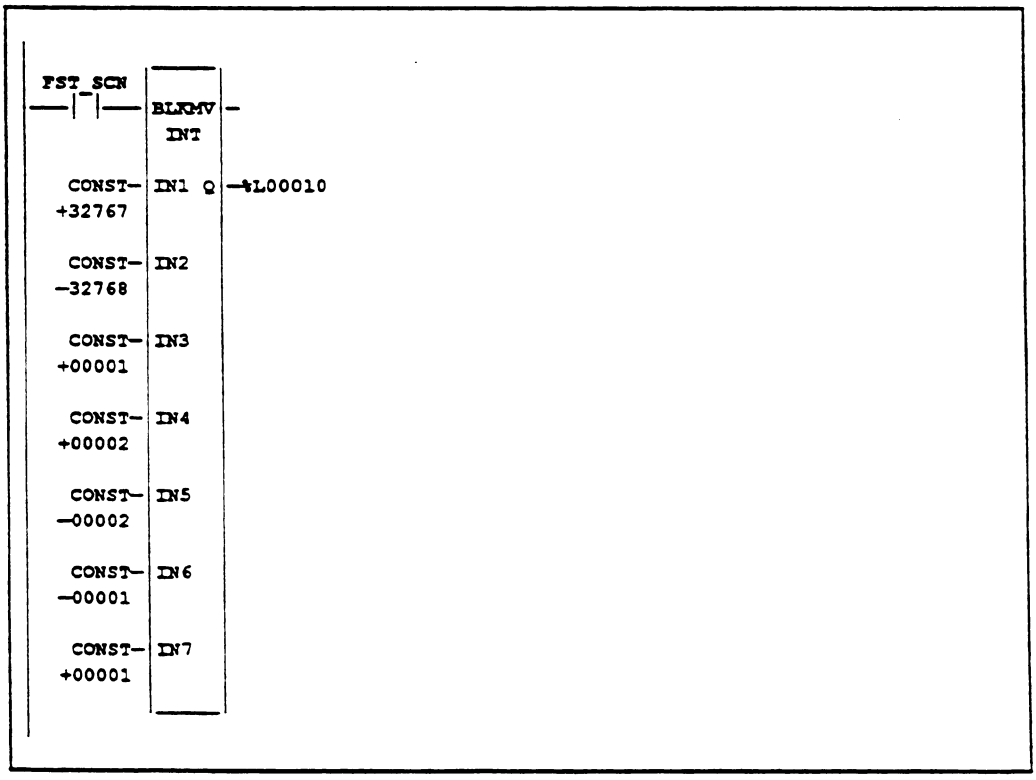
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN1-IN7													•	
ok	•													•
Q		o	o	o	o	o†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für INT, UINT oder WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Bei dem folgenden Beispiel kopiert die BLKMV-Funktion die sieben Eingangskonstanten in die Speicheradressen %L00010 bis %L00016, wenn der durch die symbolische Adresse FST-SCN gekennzeichnete Freigabeeingang "wahr" ist.

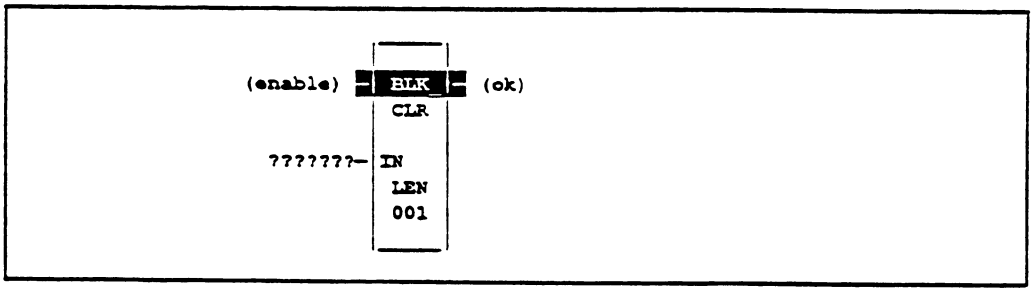


BLKCLR (WORD)

Mit der BLKCLR-Funktion kann ein spezifizierter Datenblock mit Nullen gefüllt werden.

Die BLKCLR-Funktion besitzt zwei Ein- und einen Ausgangsparameter. Erhält die Funktion Stromfluß, dann schreibt sie die Speicheradresse, die mit dem durch IN festgelegten Parameter beginnt, mit Nullen voll. Stehen die gelöschten Daten im diskreten Speicher (%I, %Q, %M, %G oder %T), dann wird die mit den diskreten Speicherelementen verbundene Transitionsinformation ebenfalls gelöscht. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die BLKCLR-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Das Feld wird gelöscht, wenn die Funktion freigegeben ist.
IN	IN enthält den ersten Wert des Feldes, das gelöscht werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist.

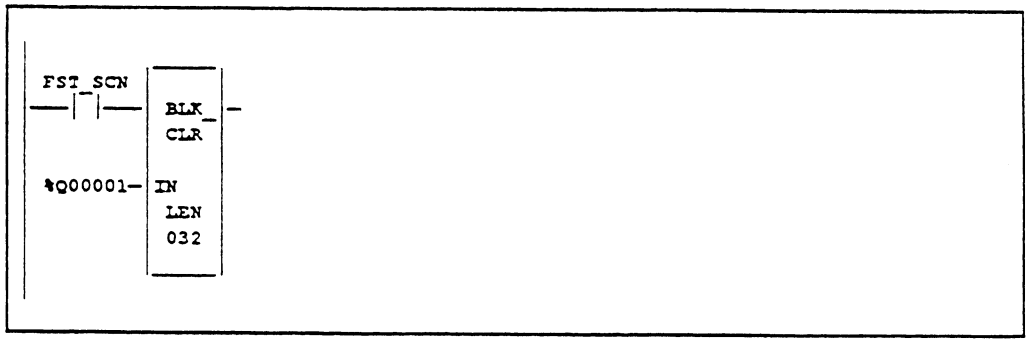
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN		o	o	o	o	o†	o	•	•	•	•	•		
ok	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im folgenden Beispiel werden beim Einschalten 32 Worte des %Q-Speichers (512 Punkte), beginnend mit %Q0001, mit Nullen gefüllt. Die mit diesen Referenzen verbundene Transitionsinformation wird ebenfalls gelöscht.



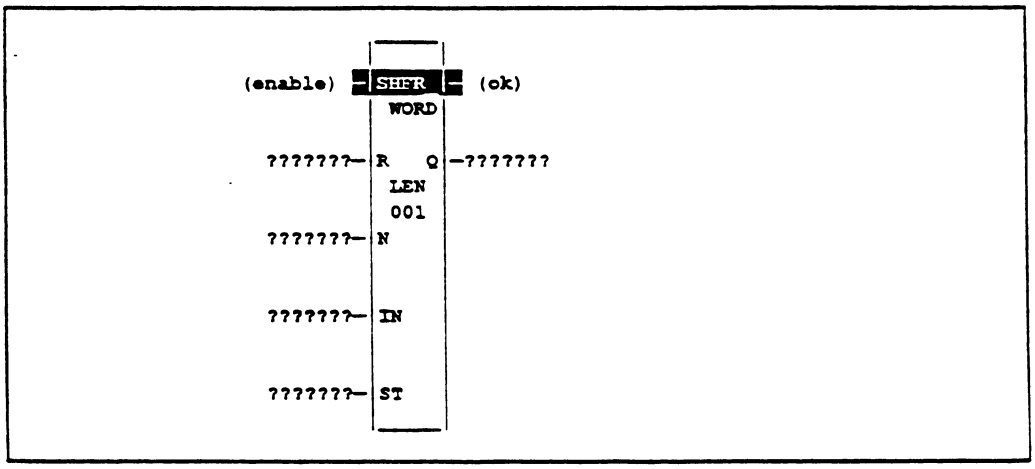
SHFREG (BIT, WORD, DWORD)

Mit der SHFREG-Funktion können ein oder mehrere Datenworte oder Datenbits von einer Referenzadresse in einen angegebenen Speicherbereich verschoben werden. So kann zum Beispiel ein Wort in einen Speicherbereich mit einer angegebenen Länge von fünf Worten geschoben werden. Als Ergebnis dieses Vorgangs wird am Ende des Speicherbereichs das letzte Datenwort herausgeschoben.

Die SHFREG-Funktion besitzt fünf Ein- und zwei Ausgangsparameter. Die Rücksetz-Freigabelogik (R) hat Priorität über die Funktionsfreigabelogik. Ist RESET aktiv, dann werden über die für LEN angegebene Länge ab dem Schieberegister (ST) alle Referenzen mit Nullen gefüllt.

Erhält die Funktion Stromfluß, ohne daß RESET aktiv ist, werden die Daten im Schieberegister um die Anzahl der in N angegebenen Elemente (Bits oder Worte) nach unten verschoben. Das letzte Element aus dem Schieberegister wird in Q eingetragen. Das Element, das in IN ganz rechts steht, wird in das bei ST beginnende geleerte Element eingetragen. Auf den Inhalt des Schieberegisters kann aus dem gesamten Programm heraus zugegriffen werden, da er in absoluten Adressen im adressierbaren Programmspeicher überlagert ist.

Die Länge des Schieberegisters wird durch LEN festgelegt. Bei SHFR_WORD muß der Wert des Operanden LEN, der die Anzahl Worte angibt, zwischen 1 und 256 liegen. Bei SHFR_BIT muß der Wert des Operanden LEN, der die Anzahl Bits angibt, zwischen 1 und 256 liegen. Die Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie über die Freigabelogik Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die Verschiebung wird durchgeführt, wenn "enable" durchgeschaltet ist und R nicht.
R	Das Schieberegister bei ST wird mit Nullen gefüllt, wenn R durchgeschaltet wird.
N	N enthält die Anzahl der Elemente, die in das Schieberegister geschoben werden sollen.
IN	IN enthält den Wert, der ins erste Bit oder Wort des Schieberegisters eingeschoben werden soll. Bei SHFREG_BIT kann jede diskrete Referenz verwendet werden, die nicht im Byteraster zu liegen braucht. Es werden jedoch ab der angegebenen Referenzadresse 16 Bits on-line angezeigt.
ST	ST enthält die Referenzadresse des ersten Bits oder Wortes des Schieberegisters. Bei SHFREG_BIT kann jede diskrete Referenz verwendet werden, die nicht im Byteraster zu liegen braucht. Es werden jedoch ab der angegebenen Referenzadresse 16 Bits on-line angezeigt.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist.
Q	Der Ausgang Q enthält die aus dem Schieberegister herausgeschobenen Elemente. Bei SHFREG_BIT kann jede diskrete Referenz verwendet werden, die nicht im Byteraster zu liegen braucht. Es werden jedoch ab der angegebenen Referenzadresse 16 Bits on-line angezeigt.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
R	•													
N													•	
IN	*	o	o	o	o	o	o	•	•	•	•	•	•	
ST		o	o	o	o	o†	o	•	•	•	•	•		
ok	•													•
Q	*	o	o	o	o	o†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

* = Nur für WORD oder DWORD.

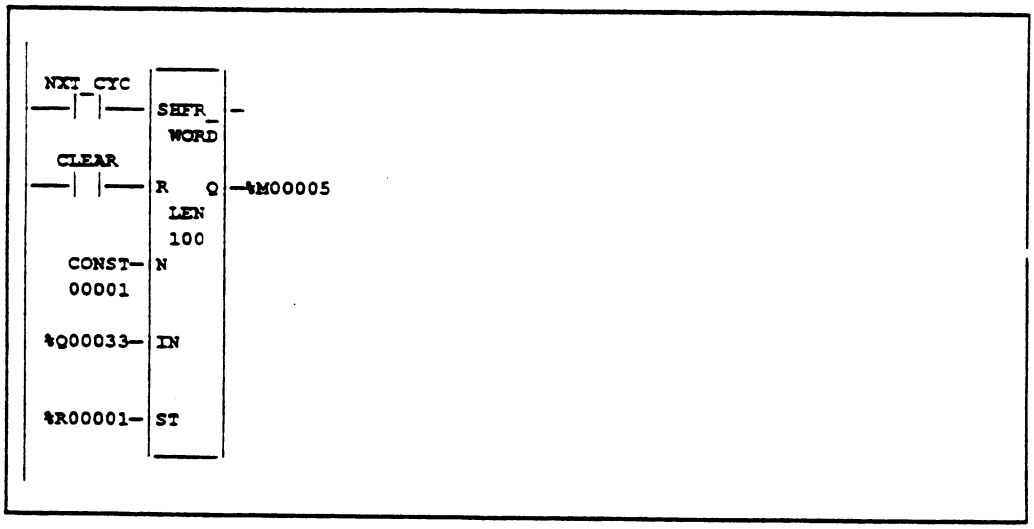
o = Datentyp nur für BIT oder WORD zulässig.
Bei SHFREG_BIT brauchen die diskreten Speichertypen %I, %Q, %M und %T nicht im Byteraster zu liegen.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel 1:

Bei dem folgenden Beispiel wird die Funktion auf die Registerspeicher-Adressen %R00001 bis %R000100 angewandt. Die Schieberegisterworte werden auf Null gesetzt, wenn die Rücksetzreferenz CLEAR aktiv ist.

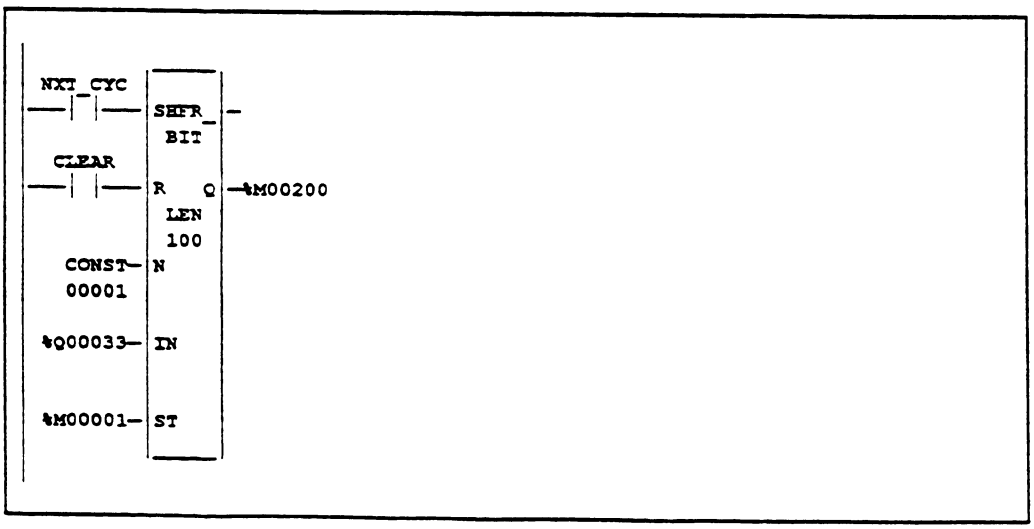
Ist die Referenz NXT_CYC aktiv und ist CLEAR nicht aktiv, dann wird das Wort aus der Ausgangsstatustabellen-Adresse %Q00033 in das Schieberegister geschoben. Das aus dem Schieberegister herausgeschobene Wort wird im Ausgang %M00005 abgelegt. Beachten Sie, daß in diesem Beispiel die für LEN angegebene Länge und die Anzahl der zu verschiebenden Daten (N) nicht gleich sind.



Beispiel 2:

In diesem Beispiel benutzt die Schieberegisterfunktion die Speicheradressen %M00001 bis %M00100. Ist die Rücksetzreferenz CLEAR aktiv, dann füllt die SHFREG-Funktion %M00001 bis %M00100 mit Nullen.

Ist NXT_CYC aktiv und CLEAR nicht aktiv, dann verschiebt die SHFREG-Funktion die Daten im Bereich %M00001 bis %M00100 um ein Bit nach unten. Das Bit aus %Q00033 wird nach %M00001 geschoben und das aus %M00100 herausgeschobene Bit wird in %M00200 eingetragen.



BITSEQ

Mit der BITSEQ-Funktion kann eine Bitfolge durch ein Bitfeld verschoben werden. Die BITSEQ-Funktion besitzt fünf Eingangs und einen Ausgangsparameter. Das Ergebnis der Funktion hängt von dem vorherigen Wert des Parameters EN ab:

R aktueller Durchlauf	EN vorheriger Durchlauf	EN aktueller Durchlauf	BITSEQ-Funktion
falsch	falsch	falsch	BITSEQ-Funktion wird nicht ausgeführt
falsch	falsch	wahr	BITSEQ-Funktion erhöht/erniedrigt um 1
falsch	wahr	falsch	BITSEQ-Funktion wird nicht ausgeführt
falsch	wahr	wahr	BITSEQ-Funktion wird nicht ausgeführt
wahr	wahr/falsch	wahr/falsch	BITSEQ-Funktion wird rückgesetzt

Der Rücksetzeingang (R) überschreibt den Freigabeeingang (EN) und bewirkt immer ein Rücksetzen der BITSEQ-Funktion. Ist der Rücksetzeingang aktiv, dann wird die aktuelle Schrittnummer auf den über den Schrittnummernparameter eingegebenen Wert gesetzt. Wurde keine Schrittnummer eingegeben, dann wird der Schritt auf 1 gesetzt. Mit Ausnahme des Bits, auf das der aktuelle Schritt zeigt und das auf 1 gesetzt wird, werden sämtliche Bits in der Bitfolgesteuerung auf 0 gesetzt.

Ist EN aktiv und R nicht, dann wird das Bit gelöscht, auf das der aktuelle Schritt zeigt. Die aktuelle Schrittnummer wird je nach Richtungsparameter entweder erhöht oder erniedrigt. Danach wird das Bit, auf das die neue Schrittnummer zeigt, auf 1 gesetzt.

- Die Schrittnummer wird auf 1 rückgesetzt, wenn sie beim Erhöhen den Bereich 1 ≤ Schrittnummer ≤ LEN verläßt.

- Die Schrittnummer wird auf LEN rückgesetzt, wenn sie beim Erniedrigen den Bereich 1 ≤ Schrittnummer ≤ LEN verläßt.

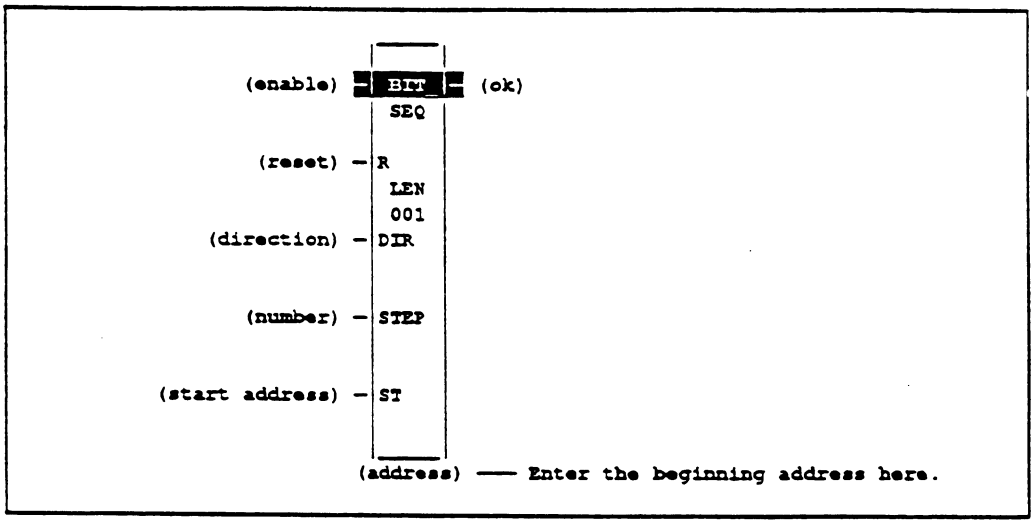
Die Angabe des START-Parameters ist nicht zwingend. Wird er nicht verwendet, dann arbeitet die BITSEQ-Funktion wie beschrieben, es werden jedoch keine Bits gelöscht oder gesetzt. Die BITSEQ-Funktion schaltet lediglich die aktuelle Schrittnummer durch den zulässigen Bereich hindurch.

Speicherbedarf der Bitfolgesteuerung

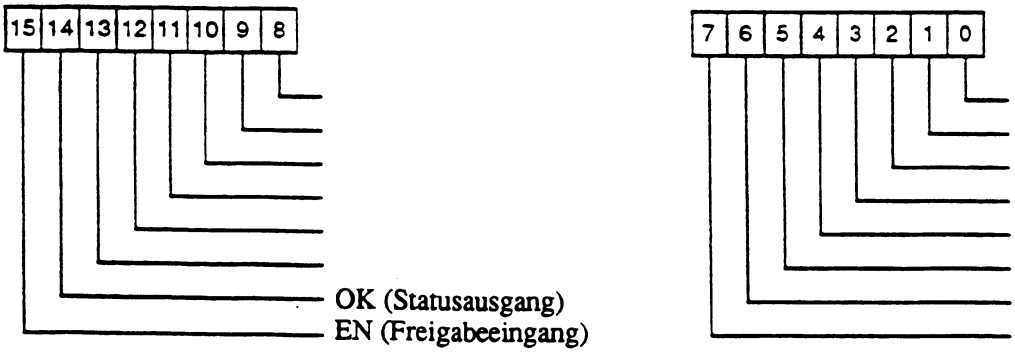
Jede Bitfolgesteuerung belegt im %R-, %L- oder %P-Speicher drei Worte, um die folgenden Daten abzuspeichern:

aktuelle Schrittnummer	Wort 1
Länge der Bitfolge (in Bits)	Wort 2
Steuerwort	Wort 3

Bei der Eingabe der BITSEQ-Funktion müssen Sie eine Anfangsadresse für diese drei Worte (Register) unmittelbar unterhalb der Funktionsgraphik eingeben. Zum Beispiel:



Im Steuerwort werden die Zustände der Boolschen Ein- und Ausgänge des zugehörigen Funktionsblocks in folgendem Format abgelegt:



Parameter:

Parameter	Beschreibung
address	Gibt die Speicheradresse von aktuellem Schritt und Länge sowie der letzten Freigabe- und OK-Zustände an.
enable	Die Bitfolge wird verschoben, wenn die Funktion freigegeben wird, nachdem sie beim letzten Zyklus nicht freigegeben war und R nicht durchgeschaltet ist.
R	Wird R durchgeschaltet, dann wird die Schrittnummer der BITSEQ-Funktion auf den Wert in STEP (Standard = 1) gesetzt und mit Ausnahme des aktuellen Schrittnummernbits werden alle Bits der Folge auf Null gesetzt.
DIR	Wird DIR durchgeschaltet, dann wird die Schrittnummer vor dem Verschieben um eins erhöht. Ist DIR nicht durchgeschaltet, dann wird sie erniedrigt.
STEP	Die Schrittnummer wird auf diesen Wert eingestellt, wenn R durchgeschaltet wird.
ST	ST enthält das erste Wort der Bitfolge.
OK	Der OK-Ausgang wird bei jeder Freigabe der Funktion durchgeschaltet.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
address								•						
enable	•													
R	•													
DIR	•													
STEP		•	•	•	•		•	•	•	•	•	•	•	•
ST		•	•	•	•	•†	•	•	•	•		•		
ok	•											•		

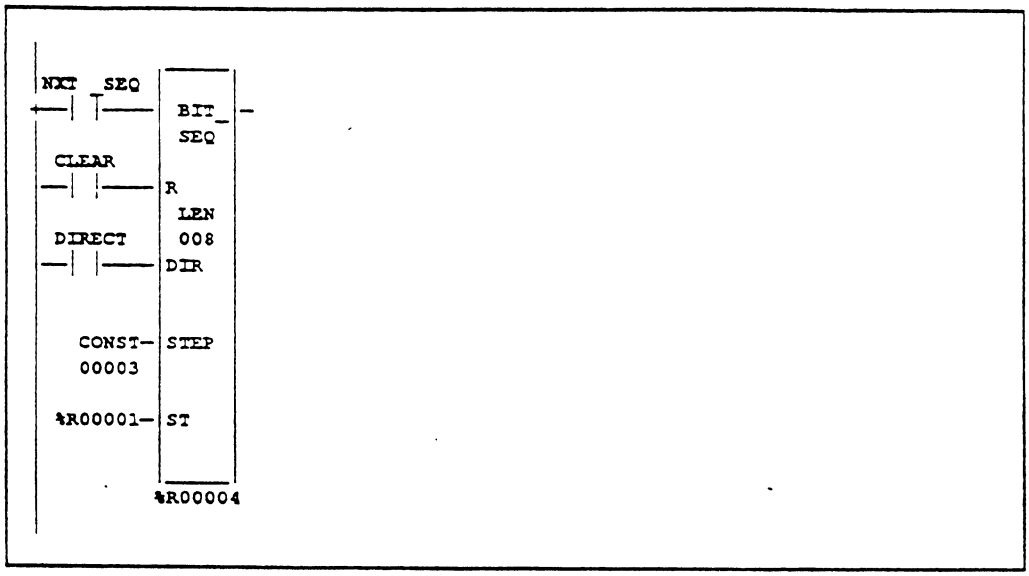
• = Datentyp nur für BIT-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Beim folgenden Beispiel verwendet die BITSEQ-Funktion den Registerspeicher %R00001. Die statischen Daten sind in den Registern R00004, R00005 und R00006 abgelegt. Ist CLEAR aktiv, dann wird die Funktion rückgesetzt und der aktuelle Schritt wird auf Schrittnummer 3 eingestellt. Die ersten acht Bits von %R00001 werden auf Null gesetzt.

Ist NXT_SEQ aktiv und CLEAR nicht, dann wird das Bit für Schrittnummer 3 gelöscht und das Bit für Schrittnummer 2 oder 4 (hängt davon ab, ob DIR durchgeschaltet ist) wird gesetzt.



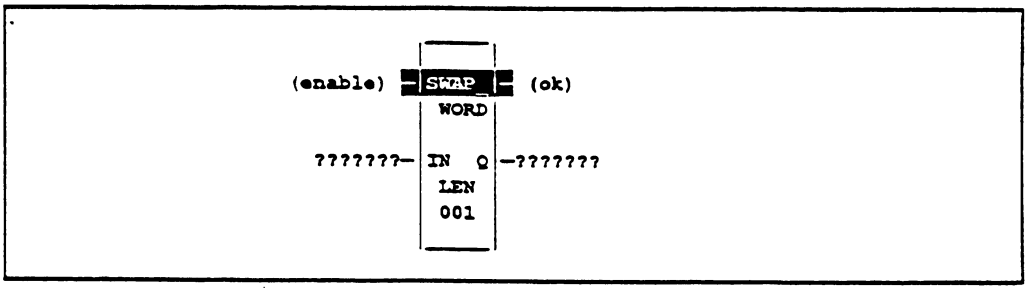
SWAP (WORD, DWORD)

Mit der SWAP-Funktion können Sie zwei Bytes innerhalb eines Wortes oder zwei Worte innerhalb eines Doppelworts gegeneinander vertauschen.

Wird eine Länge größer als 1 angegeben, dann kann die SWAP-Funktion auf einen bestimmten Speicherbereich angewandt werden. In diesem Fall wird jedes Wort oder Doppelwort innerhalb des angegebenen Bereichs entsprechend vertauscht.

Die SWAP-Funktion besitzt zwei Eingangs- und zwei Ausgangsparameter. Erhält sie Stromfluß, dann führt sie den Austausch bei jedem Wort oder Doppelwort innerhalb des angegebenen Bereichs durch. Die Ergebnisse des Austauschs werden in Ausgang Q gespeichert. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die SWAP-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die SWAP-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
IN	IN enthält die Daten, die vertauscht werden sollen.
ok	Dieser Ausgang wird durchgeschaltet, wenn der Freigabeeingang (enable) aktiviert ist.
Q	Ausgang Q enthält die vertauschte Version der Originaldaten IN.

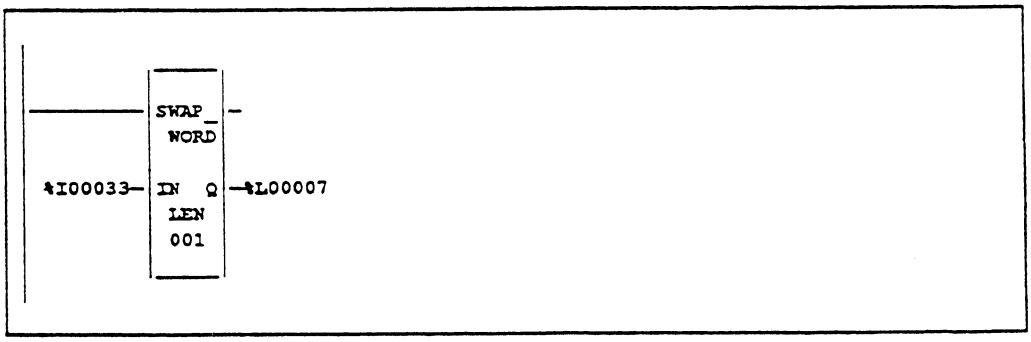
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für WORD-Daten zulässig.

Beispiel:

Im folgenden Beispiel werden die beiden Bytes in den Bits %I00033 bis %I00048 miteinander vertauscht. Das Ergebnis wird in %L00007 abgelegt.



COMMREQ

Die COMMREQ-Funktion wird dann eingesetzt, wenn das Programm mit einem intelligenten Modul (z.B. Buscontroller, programmierbares Coprozessormodul oder LAN-Schnittstellenmodul) Daten austauschen soll. Auf den folgenden Seiten wird lediglich das Format der COMMREQ-Funktion dargestellt. Für jeden Gerätetyp brauchen Sie zusätzliche Angaben bei der Programmierung der COMMREQ-Funktion, die Sie in der Gerätedokumentation zu den jeweiligen Modulen finden.

Die Kommunikationsarten, die angefordert werden können, seien hier am Beispiel des Genius-Buscontrollers aufgezeigt:

Befehl	Beschreibung
1	Ausgangs-Impulstest.
2	Konfigurationsdaten von einem Busteilnehmer lesen.
3	Konfigurationsdaten zu einem Busteilnehmer schreiben.
4	Diagnosedaten lesen.
5	Schaltkreisfehler löschen.
6	Alle Schaltkreisfehler löschen.
7	Monitor zuweisen.
8	Ausgänge freigeben/sperren.
9	Globaldaten freigeben/sperren.
10	Busumschaltmodul betätigen.
11	Gerät lesen.
12	Gerät schreiben.
13	Datagramm aus Warteschlange nehmen.
14	Datagramm senden.
15	Datagramm empfangen.

Mit den vorstehenden Befehlen 14 und 15 (Datagramm senden oder empfangen) können verschiedene andere Meldungen übertragen werden. Programmierinformationen zum Buscontroller finden Sie in GFK-0398.

Beachten Sie, daß die Teilnehmer auch während des System-Kommunikationswindows, das automatisch am Ende jedes Zyklus geöffnet wird, Globaldaten mit der SPS austauschen können. Diese Kommunikationsart benötigt keine Programmunterstützung.

Die COMMREQ-Funktion besitzt vier Eingangs- und zwei Ausgangsparameter. Erhält eine COMMREQ-Funktion Stromfluß, dann wird ein Befehlsdatenblock zu TASK geschickt. Dieser Befehlsblock beginnt mit der über den Parameter IN spezifizierten Referenz. Das Gerät, mit dem Daten ausgetauscht werden sollen, wird durch seine in SYSID angegebene Chassis- und Steckplatznummer identifiziert.

Die Kommunikations-Anforderungsfunktion sendet entweder eine Meldung und wartet auf Antwort oder sie sendet eine Meldung und fährt ohne Antwort fort. Wurde im Befehlsblock angegeben, daß das Programm nicht auf eine Antwort wartet, dann wird der Inhalt des Befehlsblocks zu dem empfangenden Teilnehmer geschickt und die Programmbearbeitung unmittelbar fortgesetzt (der Zeitüberwachungswert wird ignoriert). Diese Betriebsart bezeichnet man mit NOWAIT.

Gibt der Befehlsblock an, daß das Programm auf eine Antwort wartet, dann wird der Inhalt des Befehlsblocks zu dem empfangenden Teilnehmer gesendet und die CPU wartet auf eine Antwort. Die maximale Wartezeit entspricht dabei dem im Befehlsblock angegebenen Wert. Antwortet der Teilnehmer nicht innerhalb der vorgegebenen Zeit, dann wird die Programmbearbeitung fortgesetzt. Diese Betriebsart bezeichnet man mit WAIT.

Die Funktion schaltet Stromfluß durch, wenn die vorgegebene Zeitdauer nicht überschritten wurde (Timeout) und keine Nullzeit als Überwachungszeit angegeben wurde. Der Funktionsfehlerausgang (FT) wird auf "wahr" gesetzt, wenn

1. die angegebene Zieladresse nicht existiert;
2. die angegebene Task für das Gerät nicht zulässig ist;
3. die Datenlänge 0 beträgt;
4. die Status-Pointeradresse (Teil des Befehlsblocks) des Geräts nicht existiert. Ursache kann eine fehlerhafte Wahl des Speichertyps sein oder eine Adresse innerhalb dieses Speichertyps, die außerhalb des zulässigen Bereichs liegt.

Der FT-Ausgang kann einen der folgenden Zustände annehmen:

Freigabe	Fehler ?	FT-Ausgang
aktiv	nein	falsch
aktiv	ja	wahr
nicht aktiv	keine Ausführung	falsch

Befehlsblock

Der Befehlsblock liefert die zusätzlichen von der COMMREQ-Funktion benötigten Angaben.

Die Adresse des Befehlsblocks wird für den IN-Eingang zur COMMREQ-Funktion angegeben. Bei dieser Adresse kann es sich um einen beliebigen wortstrukturierten Speicherbereich (%P, %L, %R, %AI oder %AQ) handeln. Die Länge des Befehlsblocks hängt von der zum Gerät gesendeten Datenmenge ab.

Der Befehlsblock besitzt folgende Struktur:

Länge	Adresse
Anzeige 'warten/nicht warten'	Adresse + 1
Statuszeiger Speicher	Adresse + 2
Statuszeiger Offset	Adresse + 3
Pausenüberwachungswert	Adresse + 4
Max. Kommunikationsdauer	Adresse + 5
Datenblock	Adresse + bis Adresse + 133

Die für den Befehlsblock erforderlichen Angaben können mit der entsprechenden Programmierfunktion in die angegebenen Speicherbereiche eingetragen werden.

Bei der Eingabe der Daten in den Befehlsblock gelten die folgenden Definitionen:

Länge: Die Datenmenge zwischen Adresse + 6 und dem Ende des Befehlsblocks. Hierzu gehören auch die Befehlsnummer sowie der Datenblockanteil der COMMREQ-Funktion. Die Gesamtlänge liegt zwischen 2 (für einen Befehlsblock mit der Datenlänge 1) und 128 Worten. Jeder COMMREQ-Funktionstyp besitzt einen eindeutigen Datenblock.

Anzeige 'warten/nicht warten': Hier wird eingestellt, ob das Programm auf den Abschluß des Datenverkehrs warten soll.

Für	Eingabe
nicht warten	0
auf Antwort warten	1

Das Anzeigenbit liegt im niedrigstwertigen Bit von (Adresse +1). Der Rest der Adresse wird mit Nullen aufgefüllt. Wird der Befehlsblock mit ganzzahligen Werten programmiert, dann erfolgt dies automatisch.

Statuszeiger: Die CPU-Adresse, in die das Gerät seinen Status einträgt.

Statuszeiger Speicher	Adresse + 2
Statuszeiger Offset	Adresse + 3

Das untere Byte von (Adresse +2) enthält eine Zahl, die den Speichertyp angibt, in dem der Statuszeiger liegt.

Für diesen Speichertyp		geben Sie diese Zahl ein
%I	diskrete Eingangstabelle (Bitmodus)	70
%Q	diskrete Ausgangstabelle (Bitmodus)	72
%I	diskrete Eingangstabelle (Bytemodus)	16
%Q	diskrete Ausgangstabelle (Bytemodus)	18
%R	Registerspeicher	8
%AI	analoge Eingangstabelle	10
%AQ	analoge Ausgangstabelle	12

Das obere Byte von (Adresse +2) wird nicht verwendet.

(Adresse +3) enthält die Adresse innerhalb des gewählten Speichertyps.

Die Art, in der das Gerät seinen Status während des Datenaustauschs zur SPS übermittelt, hängt vom Gerätetyp ab.

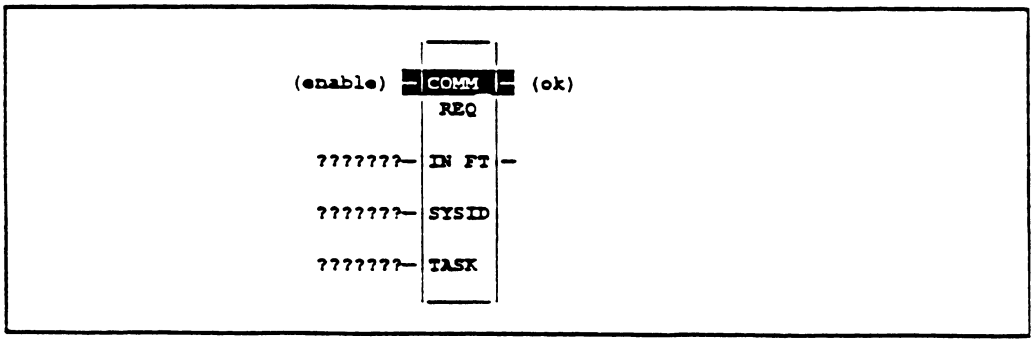
HINWEIS

Die Statuszeiger-Offsetadresse ist eine Zahl auf der Basis Null. %R00001 hat zum Beispiel die Adresse Null.

Leerlauf-Zeitüberwachung: Dieser Parameter wird im NOWAIT-Kommunikationsmodus nicht verwendet. Wurde WAIT [warten] eingestellt, dann wird in (Adresse +4) die Pausenüberwachungszeit in Schritten von 100 µs eingestellt. Die Pausenüberwachungszeit ist die maximale Zeitdauer, über die die SPS auf eine Quittierung der Kommunikationsanforderung wartet.

Maximale Kommunikationsdauer: Beim gesamten COMMREQ-Datenaustausch muß in (Adresse +5) die maximale Zeitdauer angegeben werden, über die das Programm das Fenster offenhalten muß, wenn das Zielgerät aktiv ist. Dieser Wert wird auch in Schritten von 100 µs angegeben.

Datenblock: Der Datenblock, der die Befehlsparameter enthält, beginnt mit einer Befehlsnummer in (Adresse +6). Die Befehlsnummer gibt an, welcher Kommunikationsfunktionstyp ausgeführt werden soll (die Befehlsnummern für den Genius-Buscontroller finden Sie am Anfang des COMMREQ-Abschnittes).



Parameter:

Parameter	Beschreibung
enable	Die Kommunikationsanforderung wird ausgeführt, wenn die Funktion aktiviert ist.
IN	IN enthält das erste Wort des Befehlsblocks.
SYSID	SYSID enthält die Chassisnummer (höchstwertiges Byte) und die Steckplatznummer (niedrigstwertiges Byte) des Zielgeräts.
TASK	Dieser Parameter enthält die Task-Kennung vom Prozeß des Zielgeräts.
OK	Der OK-Ausgang wird durchgeschaltet, wenn der Freigabeeingang (enable) aktiviert wurde und keine Zeitüberschreitung auftrat.
FT	FT wird durchgeschaltet, wenn die Kommunikationsanforderung erfolglos ist.

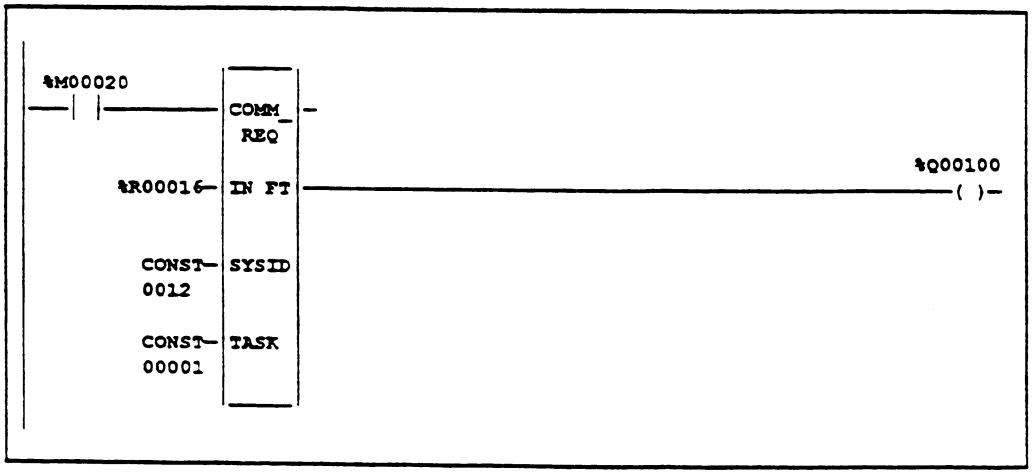
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN								•	•	•	•	•		
SYSID		•	•	•	•		•	•	•	•	•	•	•	
TASK								•	•	•	•	•	•	
OK	•													•
FT	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

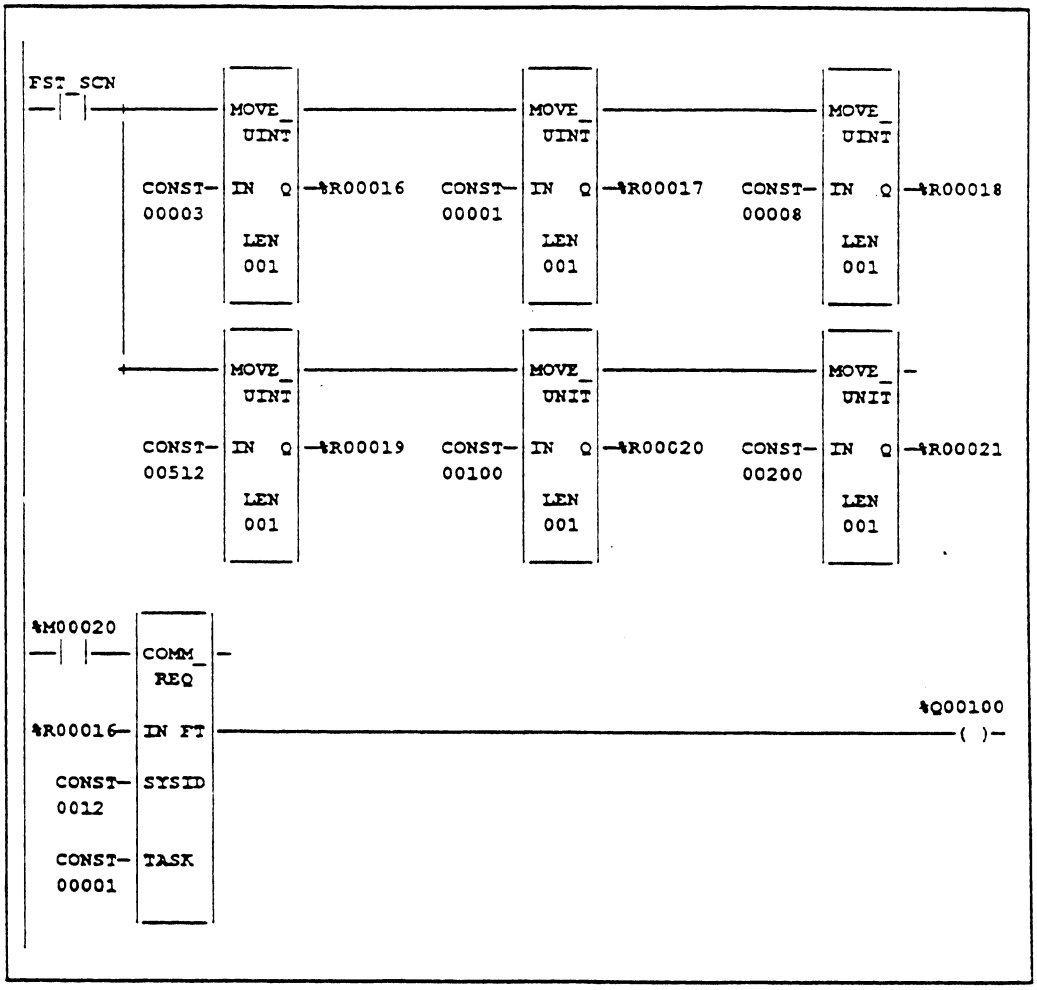
Beispiel 1:

Wird im folgenden Beispiel der Freigabeeingang %M00020 "wahr", dann wird der ab der Adresse %R00016 abgelegte Befehlsblock zur Kommunikations-Task 1 des Geräts übertragen, das in Steckplatz 2 von Chassis 1 in der SPS eingebaut ist. Tritt ein Fehler auf, dann wird %Q00100 durchgeschaltet.



Beispiel 2:

In diesem Beispiel wird mit der MOVE-Funktion der Inhalt des Befehlsblock der in Beispiel 1 beschriebenen COMMREQ-Funktion eingegeben.



Eingang IN der COMMREQ-Funktion gibt %R00016 als Anfangsadresse für den Befehlsblock an. Die nachfolgenden Referenzen enthalten folgende Daten:

Länge	%R00016
Anzeige 'warten/nicht warten'	%R00017
Statuszeiger Speicher	%R00018
Statuszeiger Offset	%R00019
Pausenüberwachungswert	%R00020
Max. Kommunikationsdauer	%R00021
Datenblock	%R00022 bis Ende der Daten

Die MOVE-Funktion liefert den folgenden Befehlsblock für die COMMREQ-Funktion. Die erste MOVE-Funktion schreibt die Länge der zu übertragenden Daten in %R00016. Die zweite MOVE-Funktion trägt die Konstante 1 in %R00017 ein, wodurch der WAIT-Modus eingestellt wird.

Die dritte MOVE-Funktion trägt die Konstante 8 in %R00018 ein, wodurch die Registertabelle als Ort des Statuszeigers festgelegt wird. Die nächste MOVE-Funktion plaziert die Konstante 512 in Referenz %R00019. Der Statuszeiger liegt dann in %R00512. Weitere MOVE-Funktionen schreiben die Konstanten 100 in %R00020 und 200 in %R00021. 100 entspricht einer Pausenwartezeit von 10 Millisekunden; 200 gibt eine maximale Datenübertragungsdauer von 20 Millisekunden an.

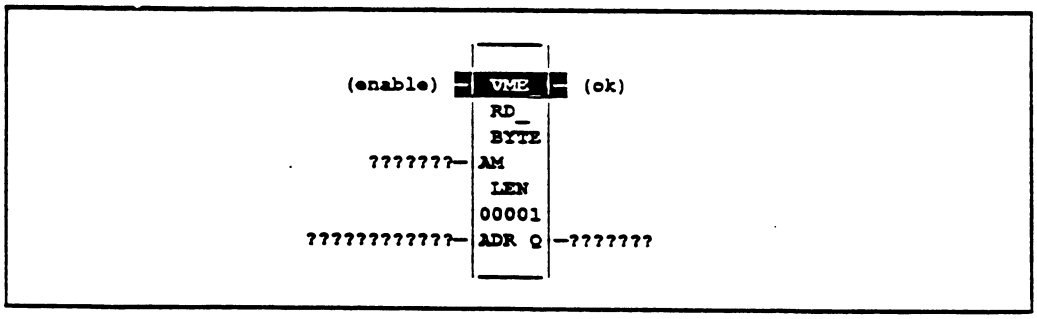
VMERD (BYTE, WORD)

Mit der VMERD-Funktion werden Daten vom VME-Bus gelesen.

HINWEIS

Die Verwendung einer VME-Funktion (VMERD, VMEWRT, VMERMV oder VMETS) setzt zusätzliche Kenntnisse über die richtige Adressierung des VME-Moduls voraus. Diese Kenntnisse können aus zwei Quellen erworben werden. Bei qualifizierten VME-Modulen können Sie normalerweise vom Vertreiber des Moduls Applikationshinweise zum richtigen Einsatz des Moduls erhalten. Ist dies nicht der Fall, dann finden Sie entsprechende Hinweise in GFK-0448.

Die VME-Lesefunktion besitzt drei Eingangs- und einen Ausgangsparameter. Erhält die Funktion Stromfluß, dann greift sie über die durch ADR und den Adreßmodifikator AM angegebene Adresse auf das VME-Modul zu und kopiert Daten mit der Länge LEN in den SPS-Speicherbereich, der bei der Ausgangsreferenz Q beginnt. Der Wert von LEN muß im Bereich zwischen 1 und 32.767 liegen. Wird die VME-Lesefunktion erfolgreich ausgeführt, dann schaltet sie den Stromfluß nach rechts durch.



Parameter:

Parameter	Beschreibung
enable	Die VME-Lesefunktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
AM	AM enthält den Adreßmodifikator.
ADR	ADR enthält die Adresse der zu lesenden Daten.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist und die Daten erfolgreich gelesen wurden.
Q	Der Ausgang Q enthält die von der durch ADR und AM festgelegten Adresse gelesenen Daten.

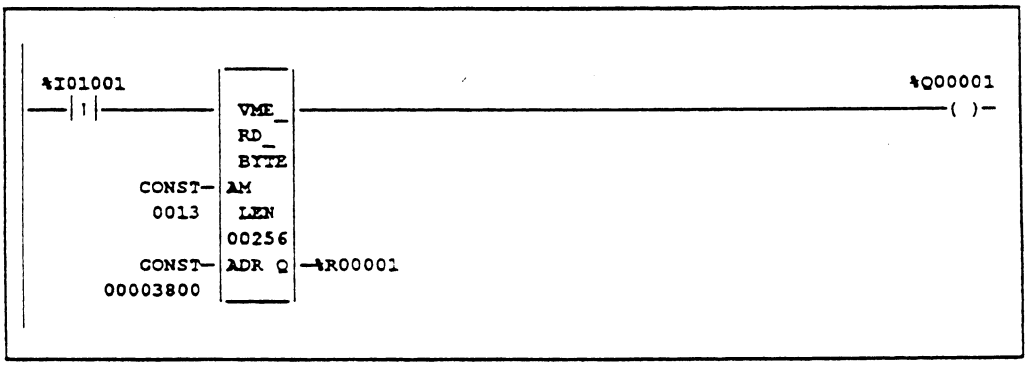
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
AM	•							•	•	•	•	•	•	
ADR	•							•	•	•	•	•	•	
ok	•													•
Q	o	o	o	o	o		o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für BYTE-Daten zulässig.

Beispiel:

Wird im folgenden Beispiel der Freigabeeingang %I01001 aktiviert, dann werden 256 Bytes von Adresse 3800 des VME-Busses in die Register %R00001 bis %R00128 gelesen. (Bei einem System mit mehreren Chassis und einem BTM entspricht dies Chassis 4). Die Spule %Q00001 wird durchgeschaltet, wenn beim Lesen der Daten kein Fehler auftritt.



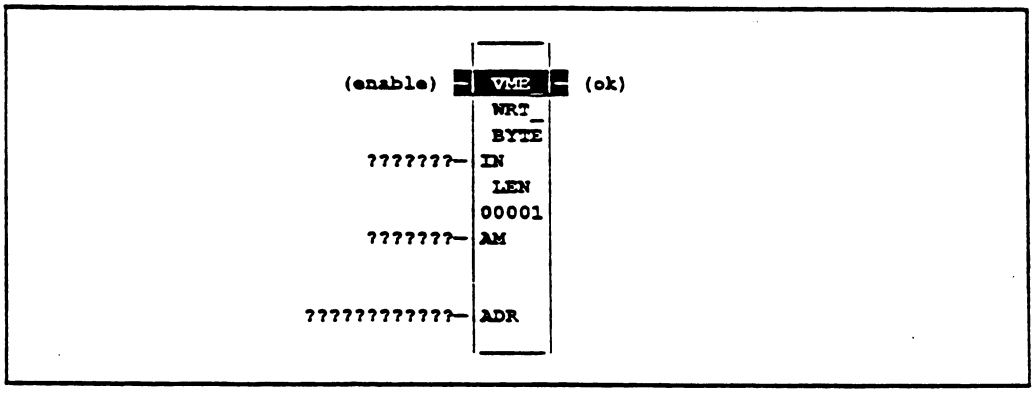
VMEWRT (BYTE, WORD)

Mit der VMEWRT-Funktion werden Daten zum VME-Bus geschrieben.

HINWEIS

Die Verwendung einer VME-Funktion (VMERD, VMEWRT, VMERMV oder VMETS) setzt zusätzliche Kenntnisse über die richtige Adressierung des VME-Moduls voraus. Diese Kenntnisse können aus zwei Quellen erworben werden. Bei qualifizierten VME-Modulen können Sie normalerweise vom Vertreiber des Moduls Applikationshinweise zum richtigen Einsatz des Moduls erhalten. Ist dies nicht der Fall, dann finden Sie entsprechende Hinweise in GFK-0448.

Die VME-Schreibfunktion besitzt vier Eingangs- und einen Ausgangsparameter. Erhält die Funktion Stromfluß, dann kopiert sie die Daten vom Eingangsparameter IN zu der durch ADR und den Adressenmodifikator AM angegebene Adresse des VME-Moduls. Der Wert von LEN muß im Bereich zwischen 1 und 32.767 liegen. Wird die VME-Schreibfunktion erfolgreich ausgeführt, dann schaltet sie den Stromfluß nach rechts durch.



Parameter:

Parameter	Beschreibung
enable	Die VME-Schreibfunktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
IN	IN enthält die Daten, die zu der durch ADR und AM angegebenen Adresse übertragen werden sollen.
AM	AM enthält den Adreßmodifikator.
ADR	ADR enthält die Adresse, in die die Daten geschrieben werden sollen.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist und die Daten erfolgreich geschrieben wurden.

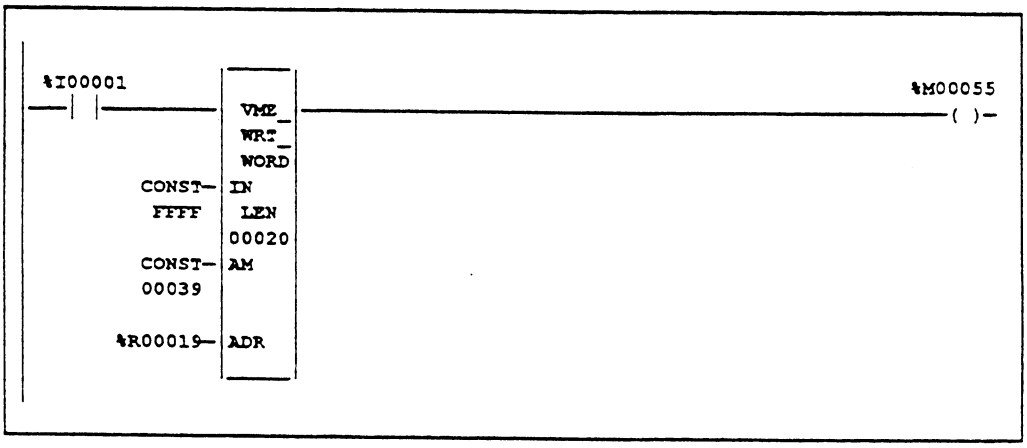
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
AM	•	•	•	•	•		•	•	•	•	•	•	•	
ADR	•							•	•	•	•	•	•	
ok	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für BYTE-Daten zulässig.

Beispiel:

Wird im folgenden Beispiel der Freigabeeingang %M01001 aktiviert, dann wird der Hexadezimalwert FFFF in jedes der 20 Worte am VME-Bus eingetragen. Das erste (niedrigste Adresse) Wort wird dabei durch den Inhalt von %R00019 (unteres Wort) und %R00020 (oberes Wort) angegeben. Die Referenz %M00055 wird durchgeschaltet, wenn beim Schreiben der Daten kein Fehler auftritt.



VMERMW (BYTE, WORD)

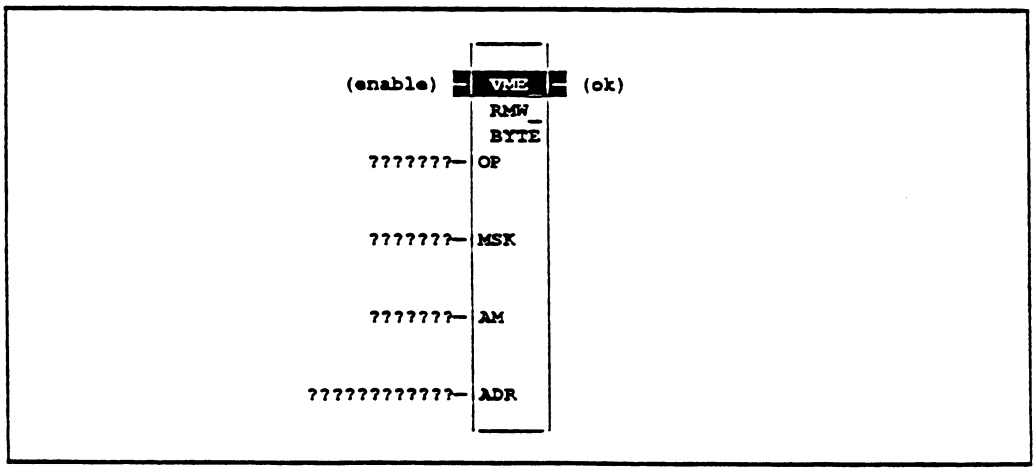
Mit der VMERMW-Funktion kann ein Datenelement am VME-Bus aktualisiert werden.

HINWEIS

Die Verwendung einer VME-Funktion (VMERD, VMEWRT, VMERMV oder VMETS) setzt zusätzliche Kenntnisse über die richtige Adressierung des VME-Moduls voraus. Diese Kenntnisse können aus zwei Quellen erworben werden. Bei qualifizierten VME-Modulen können Sie normalerweise vom Vertreiber des Moduls Applikationshinweise zum richtigen Einsatz des Moduls erhalten. Ist dies nicht der Fall, dann finden Sie entsprechende Hinweise in GFK-0448.

Die VME-Funktion zum Lesen/Ändern/Schreiben besitzt fünf Eingangs- und einen Ausgangsparameter. Erhält die Funktion Stromfluß, dann liest sie die Daten an der durch ADR und den Adressenmodifikator AM angegebene Adresse vom Modul. Dieses Datenbyte oder -Wort wird mit der Datenmaske MSK verknüpft (AND/OR). Die Wahl zwischen AND und OR wird über den Eingang OP getroffen. MSK ist ein wortstrukturierter Wert. Werden bytestrukturierte Daten verwendet, dann werden nur die unteren acht Bytes von MSK benutzt.

Das Ergebnis wird dann zur gleichen VME-Adresse zurückgeschrieben, aus der die Daten gelesen wurden. Wird die VME-Funktion zum Lesen/Ändern/Schreiben erfolgreich ausgeführt, dann schaltet sie den Stromfluß nach rechts durch.



Parameter:

Parameter	Beschreibung
enable	Die VME-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
OP	Dieser Eingang gibt an, ob die Daten über AND oder OR mit MSK verknüpft werden.
MSK	MSK enthält die Datenmaske.
AM	AM enthält den Adreßmodifikator.
ADR	ADR enthält die Adresse, in der die Daten enthalten sind, die gelesen, verändert und geschrieben werden sollen.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben und erfolgreich ausgeführt wurde.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
OP													•	
MSK	•	•	•	•	•		•	•	•	•	•	•	•	
AM	•	•	•	•	•		•	•	•	•	•	•	•	
ADR								•	•	•	•	•	•	
ok	•													•

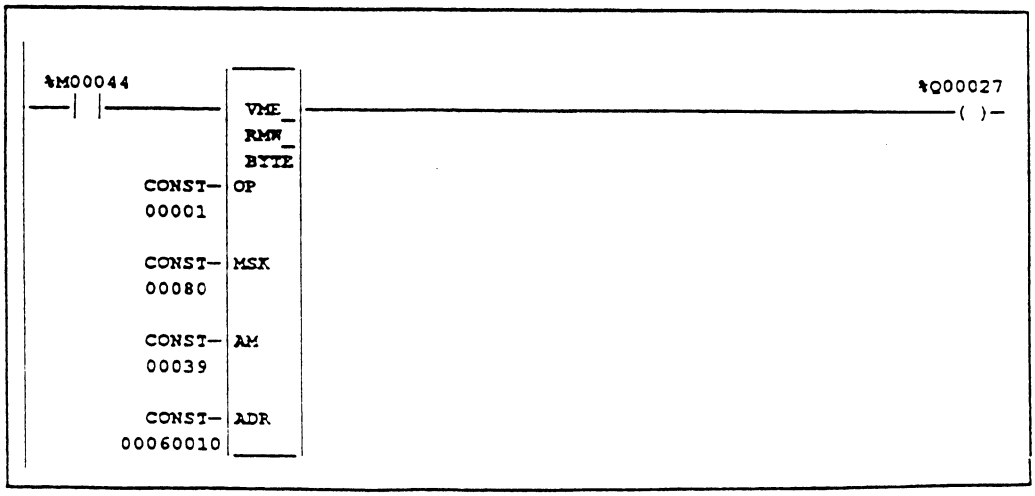
Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Wird im folgenden Beispiel der Freigabeeingang %M00044 aktiviert, dann wird der Hexadezimalwert 80H über eine OR-Funktion mit dem Datenbyte aus der Adresse 060010H am VME-Bus in Chassis 0 (dem Hauptchassis) mit nicht privilegiertem Standard-Datenzugriff verknüpft. Die Spule %Q00027 wird durchgeschaltet, wenn beim Datenzugriff kein Fehler aufgetreten ist.



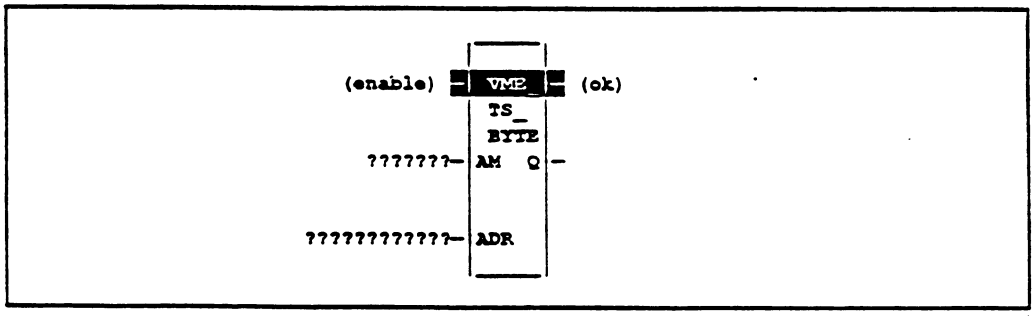
VMETS (BYTE, WORD)

Mit der VMETS-Funktion können Semaphore am VME-Bus bearbeitet werden. Die VME-Test- und Setzfunktion tauscht den aktuellen Wert an der Semaphoradresse gegen "1" aus. War dieser Wert bereits "1", dann erhält die Funktion das Semaphor nicht. War der Wert "0", dann wird das Semaphor neu gesetzt und die VME-Test- und Setzfunktion erhält das Semaphor und kann damit den von ihm gesteuerten Speicherbereich verwenden. Das Semaphor wird mit der VME-Schreibfunktion gelöscht, indem eine 0 an der Semaphoradresse eingetragen wird.

HINWEIS

Die Verwendung einer VME-Funktion (VMERD, VMEWRT, VMERMV oder VMETS) setzt zusätzliche Kenntnisse über die richtige Adressierung des VME-Moduls voraus. Diese Kenntnisse können aus zwei Quellen erworben werden. Bei qualifizierten VME-Modulen können Sie normalerweise vom Vertreiber des Moduls Applikationshinweise zum richtigen Einsatz des Moduls erhalten. Ist dies nicht der Fall, dann finden Sie entsprechende Hinweise in GFK-0448.

Die VME-Test- und Setzfunktion besitzt drei Eingangs- und zwei Ausgangsparameter. Erhält die Funktion Stromfluß, dann tauscht sie an der durch ADR und den Adressenmodifikator AM angegebenen Adresse ein Boolsches "wahr" gegen die dort abgelegten Daten aus. Die VME-Test- und Setzfunktion setzt Ausgang Q auf "1", wenn das Semaphor verfügbar ("falsch") war und übergeben wurde. Wird die VME-Test- und Setzfunktion erfolgreich ausgeführt, dann schaltet sie den Stromfluß nach rechts durch.



Parameter:

Parameter	Beschreibung
enable	Die VME-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
AM	AM enthält den Adreßmodifikator.
ADR	ADR enthält die Adresse des Semaphors.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben und fehlerfrei ausgeführt wurde.
Q	Ausgang Q ist "wahr", wenn das Semaphor verfügbar ("falsch") war. Wenn nicht, wird Q "falsch".

Zulässige Speichertypen:

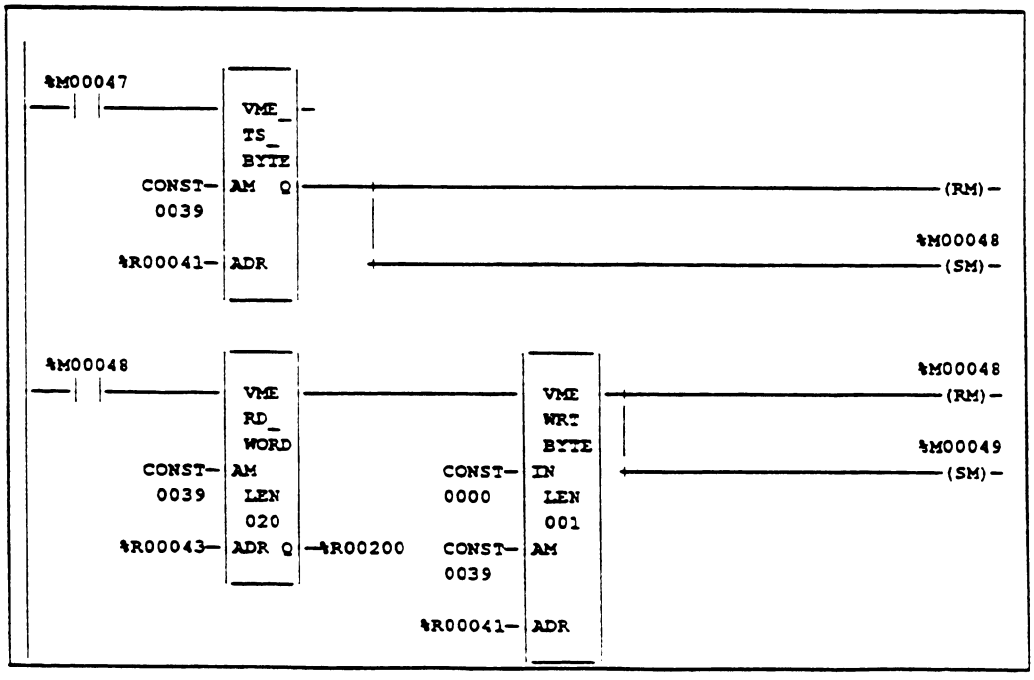
Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
AM	•							•	•	•	•	•	•	
ADR	•							•	•	•	•	•	•	
ok	•													•
Q	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Im folgenden Beispiel werden die Funktionen VMERD, VMEWRT und VMETS dazu verwendet, von einem Semaphor geschützte Daten zu lesen. Wird der Freigabeeingang %M00047 aktiviert, dann wird die VME-Test- und Setzfunktion ausgeführt, um das Semaphor zu erhalten. Die Semaphoradresse wird mit nicht privilegiertem Standard-Datenzugriff in %R00041 und %R00042 gespeichert. Ist dies erfolgreich, dann wird Merker %M00047 rückgesetzt und Merker %M00048 gesetzt. Ist %M00048 gesetzt, dann liest die VME-Funktion die Daten (20 Datenworte, deren VME-Adresse in %R00043 und %R00044 hinterlegt ist; die Daten werden in %R00200 bis %R00219 eingelesen). War der Lesevorgang erfolgreich (im anderen Fall liegt ein Defekt oder Programmierfehler vor), dann tritt die VMT-Schreibfunktion das Semaphor ab. Merker %M00048 wird rückgesetzt, wenn die VME-Schreibfunktion erfolgreich durchgeführt wurde. %M00049 wird gesetzt und zeigt an, daß nun neue Daten verfügbar sind.

VMERD und VMEWRT werden nicht ausgeführt, wenn das Semaphor nicht verfügbar ist. Wird %M00047 gesetzt, dann überprüft die SPS das Semaphor solange bei jedem Zyklus, bis es verfügbar ist. Ist das Semaphor verfügbar, wird es übernommen, die Daten werden gelesen und das Semaphor wird abgetreten. Bis %M00047 wieder gesetzt wird, werden keine weiteren Aktionen durchgeführt.



KAPITEL 9

Mit den Datentabellenfunktionen können Sie Werte in eine Tabelle eintragen oder Werte aus einer Tabelle herauskopieren. Sämtliche Werte in der Tabelle müssen vom gleichen Typ sein: ganze Zahl mit Vorzeichen (INT), doppeltgenaue ganze Zahl (DINT), ganze Zahl ohne Vorzeichen (UINT), Wort (WORD) oder Doppelwort (DWORD).

HINWEIS

Obwohl für die Datentabellenfunktionen kein REAL-Datentyp existiert, können reelle Daten mit dem DWORD-Datentyp bearbeitet werden.

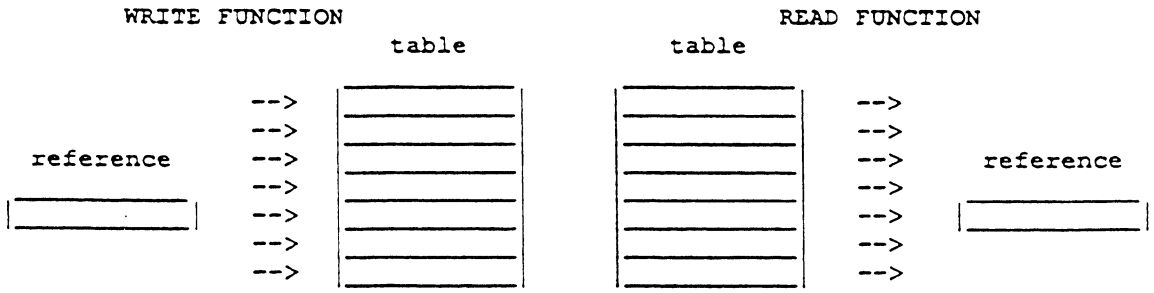
Maximal sind bei einer Datentabellenfunktion 256 Elemente möglich, lediglich die SORT-Funktion ist auf 64 Elemente beschränkt.

Programmlogik für Datentabellen

Datentabellen bieten die Möglichkeit, Daten automatisch zu bewegen. Damit diese Funktionen richtig eingesetzt werden können, muß das Programm andere Logikelemente enthalten, mit denen die Daten durch die Tabellen bewegt und Tabellenpointer entsprechend nachstehender Beschreibung eingerichtet werden können.

Datenverschiebung zu und von einer Tabelle

Sämtliche Tabellen-Schreibfunktionen füllen eine Tabelle mit Werten, die aus einer angegebenen Referenzadresse gelesen werden. Entsprechend kopieren alle Tabellen-Lesefunktionen die Werte aus einer Tabelle zu einer angegebenen Referenzadresse.



Bei jedem Aufruf dieser Funktion wird ein Wert geschrieben oder gelesen. Über andere Logikelemente im Programm müssen neue Werte in die Referenz einer Schreibfunktion eingetragen oder Werte aus der Referenz einer Lesefunktion erfaßt werden.

Einrichten des Tabellenpointers

Die Lese- und Schreibfunktionen benutzen zur Verfolgung der aktuellen Tabellenadresse einen Pointer. Dieser Wert muß über zusätzliche Programmlogik eingerichtet werden, so daß die Funktion an der richtigen Stelle mit dem Lesen oder Schreiben der Tabelle beginnen kann. Normalerweise wird der Wert in dieser Referenz mit 0 eingerichtet.

Verwendung der Datentabellenfunktionen

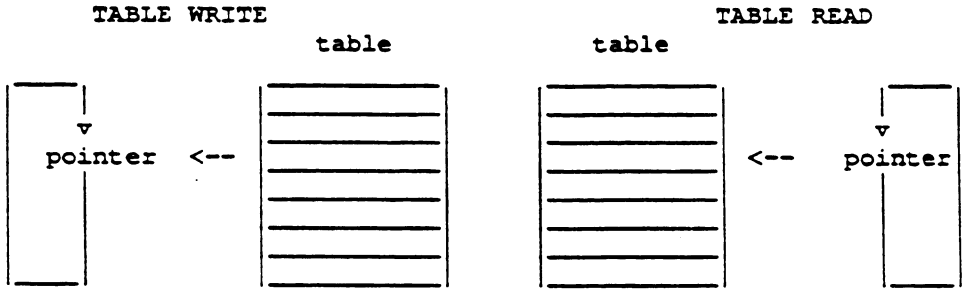
Die Tabellen-Lese- und Schreibfunktionen können als drei Paare kompatibler Funktionen betrachtet werden. Jedes Paar einer Lese-/Schreibfunktion hat einen besonderen Verwendungszweck, der nachstehend beschrieben wird.

Tabelle lesen/Tabelle schreiben

Mit diesen Funktionen können Sie Werte in einer Tabelle sequentiell aktualisieren oder lesen, ohne daß die Tabelle jemals voll wird.

- Die Tabellen-Schreibfunktion erhöht den Tabellenpointer um 1 und trägt dann einen Wert an der neuen Pointeradresse ein.

- Die Tabellen-Lesefunktion erhöht den Tabellenpointer um 1 und liest dann den Wert an der neuen Pointeradresse.



Bei beiden Funktionen geht der Pointer automatisch auf den Anfang der Tabelle zurück, wenn er das Tabellenende erreicht hat.

Die Tabellenlese- und Schreibfunktionen können die gleiche Tabelle oder unterschiedliche Tabellen benutzen. Wird für den Pointer eine unterschiedliche Referenz angegeben, dann können die beiden Funktionen die gleiche Datentabelle bei unterschiedlichen Adressen oder in unterschiedlichem Rhythmus ansprechen.

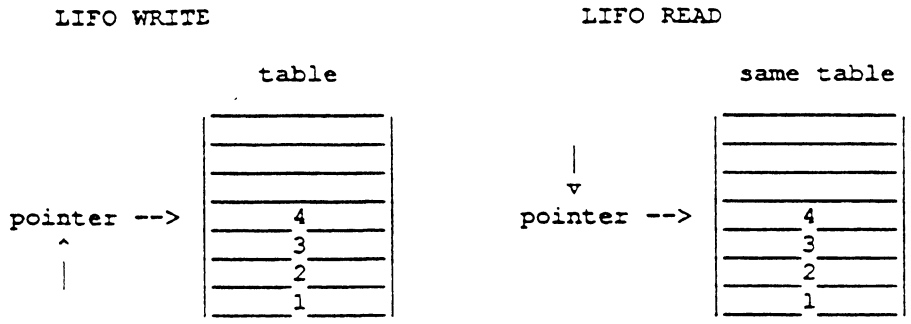
LIFOWRT/LIFORD

Mit den LIFO-Lese- und Schreibfunktionen werden Daten aus Tabellen herausgenommen oder dort eingetragen. Die Werte werden immer vom Anfang der Tabelle eingetragen oder weggenommen.

- Die LIFOWRT-Funktion erhöht den Tabellenpointer um 1 und fügt dann einen Eintrag über der Pointeradresse hinzu.

- Die LIFORD-Funktion entfernt den Eintrag bei der Pointeradresse und erniedrigt den Pointer um 1.

Werden beide Funktionen zusammen verwendet, dann ergibt sich eine Datenverwaltung in LIFO-Tabellen.



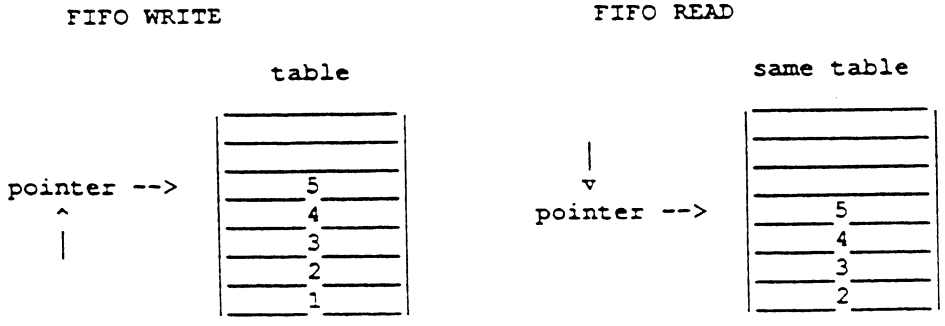
Ist die Tabelle voll (der Pointer hat die letzte Adresse erreicht), dann kann die LIFOWRT-Funktion keine weiteren Werte mehr hinzufügen. Über die Programmlogik (normalerweise die LIFORD-Funktion) müssen nun Tabelleneinträge entfernt und die Pointerposition erniedrigt werden.

FIFOWRT/FIFORD

Mit den FIFO-Lese- und Schreibfunktionen werden Daten "unten" aus Tabellen herausgenommen oder dort eingetragen.

Die FIFOWRT-Funktion erhöht den Tabellenpointer um 1 und fügt dann einen Eintrag an der neuen Pointeradresse hinzu.

Die FIFORD-Funktion entfernt den untersten Wert in der Tabelle, verschiebt die restlichen Eintragungen um eins nach unten und dekrementiert dann den Pointer. Werden diese Funktionen zusammen verwendet, dann ergibt sich eine Datenverwaltung in FIFO-Tabellen.



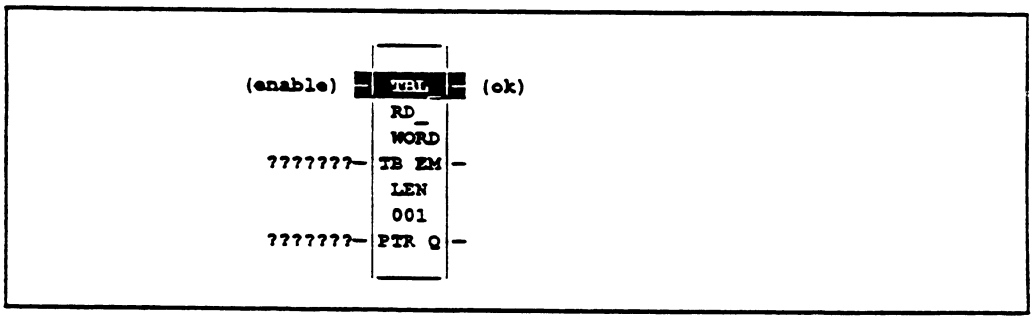
Ist die Tabelle voll (der Pointer hat die letzte Adresse erreicht), dann kann die FIFOWRT-Funktion keine weiteren Werte mehr hinzufügen. Über die Programmlogik (normalerweise die FIFORD-Funktion) müssen nun Tabelleneinträge entfernt und die Pointerposition erniedrigt werden.

TBLRD (INT, DINT, UINT, WORD, DWORD)

Mit der TBLRD-Funktion können Sie einen Wert aus einer angegebenen Tabellenposition zu einer Ausgangsreferenz kopieren. Zur Erfassung der Daten aus der Ausgangsreferenz sind zusätzliche Programmelemente erforderlich.

Die Tabellenlesefunktion besitzt drei Eingangs- und drei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann wird der Pointer (PTR) um eins erhöht. Ist diese neue Pointerposition das letzte Tabellenelement, dann wird der Ausgang EM auf "wahr" gesetzt. Bei der nächsten Ausführung der Funktion wird der Pointer automatisch zurück auf 1 gesetzt. Nachdem der Pointer erhöht wurde, wird der Inhalt an der neuen Pointerposition zur Ausgangsreferenz Q kopiert. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die TBLRD-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die TBLRD-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
TB	TB enthält die Elemente in der Tabelle.
PTR	PTR wird erhöht und zeigt dann auf das nächste Tabellenelement, das gelesen werden soll.
ok	Dieser Ausgang wird durchgeschaltet, wenn der Freigabeeingang (enable) aktiviert ist.
EM	Der Ausgang EM wird durchgeschaltet, wenn das letzte Tabellenelement gelesen wird.
Q	Ausgang Q enthält das aus der Tabelle gelesene Element.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB	•	o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
ok	•													•
EM	•													•
Q	•	o	o	o	o	Δ†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

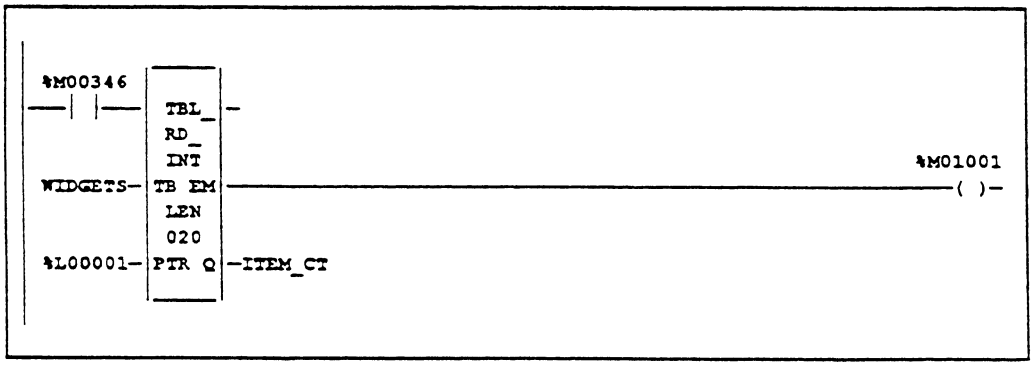
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.

Δ = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im nachstehenden Beispiel ist WIDGETS eine Tabelle mit 20 ganzzahligen Elementen. Wird der Freigabeeingang %M00346 durchgeschaltet, dann wird der Pointer erhöht und der Inhalt des nächsten Tabellenelements wird zu ITEM_CT kopiert. %L00001 arbeitet dabei als Tabellenpointer. Mit %M01001 wird signalisiert, daß auf alle Tabellenelemente zugegriffen wurde.

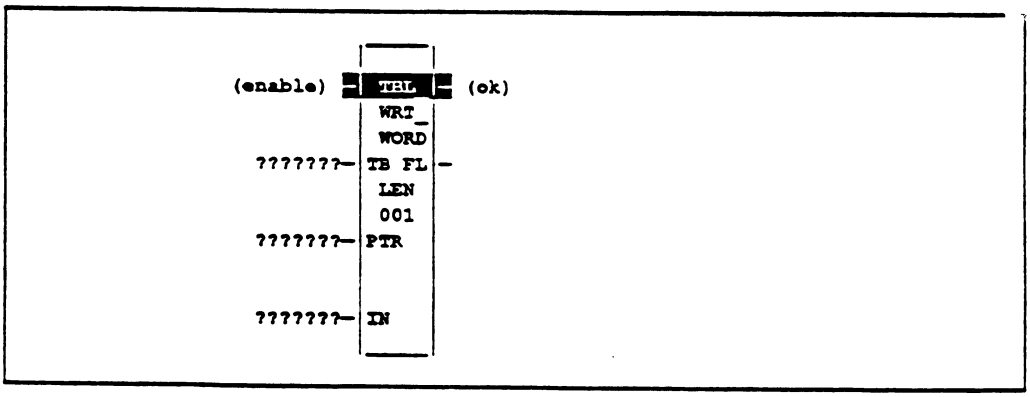


TBLWRT (INT, DINT, UINT, WORD, DWORD)

Mit der TBLWRT-Funktion können Sie einen Wert aus einer Eingangsreferenz zu einer angegebenen Tabellenposition kopieren. Sämtliche Werte an dieser Stelle werden dabei überschrieben. Die Daten müssen durch zusätzliche Programmelemente in die Eingangsreferenz eingetragen werden.

Die Tabellenschreibfunktion besitzt vier Eingangs- und zwei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann wird der Pointer (PTR) um eins erhöht. Ist diese neue Pointerposition das letzte Tabellenelement, dann wird der Ausgang FL auf "wahr" gesetzt. Bei der nächsten Ausführung der Funktion wird der Pointer automatisch zurück auf 1 gesetzt. Nachdem der Pointer erhöht wurde, schreibt die TBLWRT-Funktion den Inhalt der Eingangsreferenz in die aktuellen Pointerposition, wobei hier bereits vorhandene Daten überschrieben werden. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die TBLWRT-Funktion schaltet den Stromfluß jedesmal dann nach rechts durch, wenn sie Stromfluß erhält.



Parameter:

Parameter	Beschreibung
enable	Die TBLWRT-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
TB	TB enthält die Elemente in der Tabelle.
PTR	PTR wird erhöht und zeigt dann auf das nächste Tabellenelement, das geschrieben werden soll.
IN	IN enthält das Element, das in die Tabelle eingetragen werden soll.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist.
FL	Der Ausgang FL wird durchgeschaltet, wenn IN in das letzte Tabellenelement eingetragen ist.

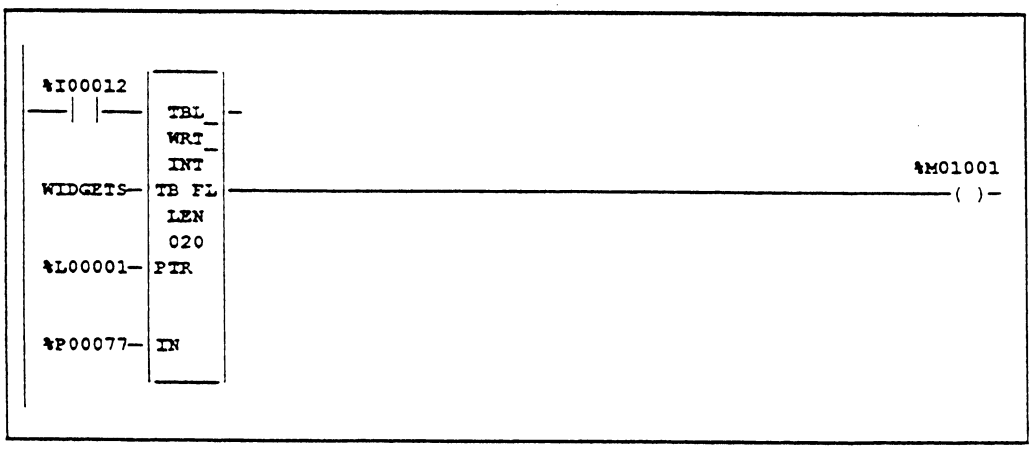
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB		o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
IN	•	o	o	o	o	Δ†	o	•	•	•	•	•	•	
ok	•													•
FL	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.
Δ = Datentyp nur für WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im nachstehenden Beispiel ist WIDGETS eine Tabelle mit 20 ganzzahligen Elementen. Wird der Freigabeeingang %I00012 durchgeschaltet, dann wird der Pointer erhöht und der Inhalt von %P00077 wird an der Pointerposition in die Tabelle eingetragen. %L00001 arbeitet dabei als Tabellenpointer.

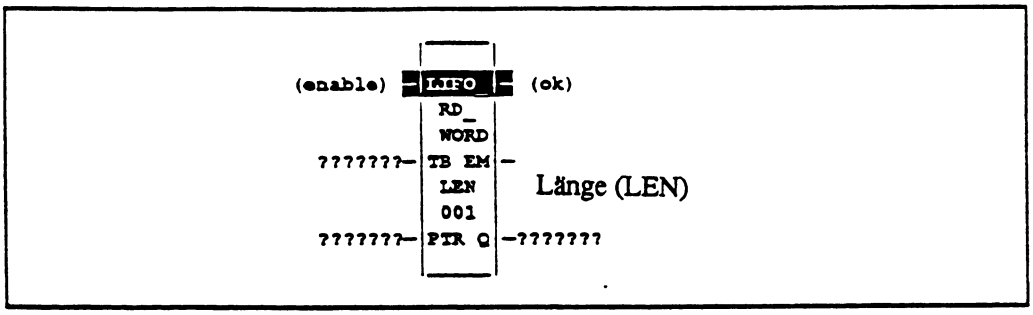


LIFORD (INT, DINT, UINT, WORD, DWORD)

Die LIFORD-Funktion wird zusammen mit der LIFO-Schreibfunktion verwendet, die den Pointer erhöht und Daten in den Stack einträgt. Die LIFORD-Funktion entfernt den Eintrag an der Pointerposition und dekrementiert den Pointer um 1. Die Daten müssen durch zusätzliche Programmelemente in die Eingangsreferenz eingetragen werden.

Die LIFO-Lesefunktion besitzt drei Eingangs- und drei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann werden die Daten an der Pointerposition zum Ausgang Q kopiert, anschließend wird der Pointer dekrementiert. Wird hierdurch die Pointerposition zu Null, dann wird der Ausgang EM "wahr". EM gibt daher an, ob die Tabelle leer ist. Ist die Tabelle leer, wenn die Funktion Stromfluß erhält, dann wird nicht gelesen. Der Pointer zeigt immer auf das letzte eingetragene Element. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die Funktion gibt Stromfluß nach rechts weiter, wenn der Pointer in einem Bereich stand, aus dem ein Element gelesen werden konnte.



Parameter:

Parameter	Beschreibung
enable	Die LIFORD-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
TB	TB enthält die Elemente im Stack.
PTR	PTR zeigt auf das nächste LIFO-Element, das gelesen werden soll. Nach dem Lesen wird der Wert dekrementiert.
ok	Dieser Ausgang wird durchgeschaltet, wenn EN "wahr" ist und $0 < PTR \leq LEN$.
EM	Der Ausgang EM wird durchgeschaltet, wenn das letzte Element gelesen wurde.
Q	Der Ausgang Q enthält das aus dem Stack gelesene Element.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB	•	o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
ok	•													•
EM	•													•
Q	•	o	o	o	o	Δ†	o	•	•	•	•	•		•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

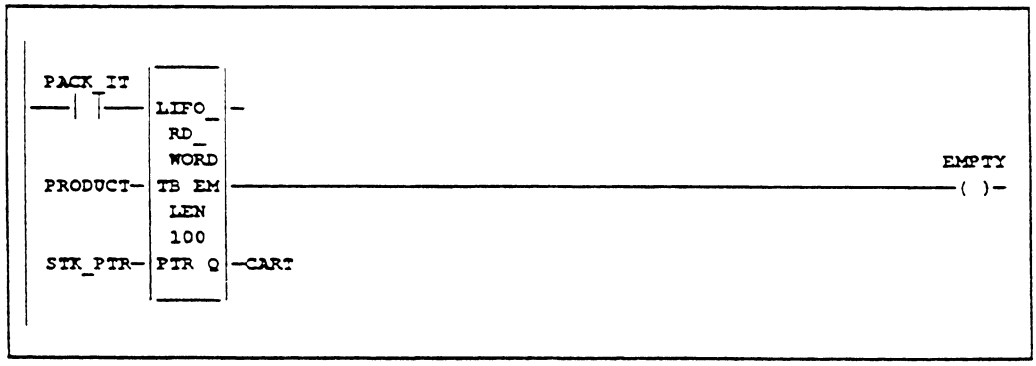
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.

Δ = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im folgenden Beispiel ist PRODUCT ein LIFO-Stack mit 100 wortstrukturierten Elementen. Ist der Freigabeeingang wahr, dann wird das oberste Datenelement im Stack in die durch die symbolische Adresse CART gekennzeichnete Referenz kopiert. Die mit STK_PTR gekennzeichnete Referenz enthält den Tabellenpointer. Die Ausgangsspule EMPTY zeigt an, daß der Stack leer ist.

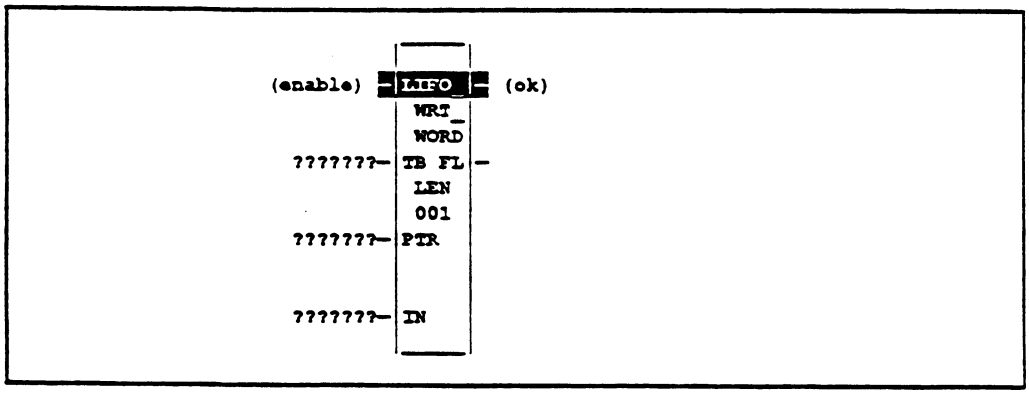


LIFOWRT (INT, DINT, UINT, WORD, DWORD)

Mit der LIFOWRT-Funktion können Sie Daten zu einer als LIFO-Stack organisierten Tabelle hinzufügen. Die LIFO-Schreibfunktion, die den Tabellenpointer erhöht und Daten in die Tabelle schreibt, wird zusammen mit der LIFORD-Funktion verwendet, die den letzten Eintrag aus dem Stack wegnimmt und den Pointer erniedrigt. Die Daten müssen durch zusätzliche Programmelemente in die Eingangsreferenz eingetragen werden.

Die LIFO-Schreibfunktion besitzt vier Eingangs- und zwei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann wird der Pointer um 1 erhöht und die neuen Daten werden an der Pointerposition eingetragen. Stand der Pointer bereits auf der letzten Tabellenposition, dann werden keine Daten eingetragen und die Funktion schaltet den Stromfluß nicht nach rechts durch. Der Pointer zeigt immer auf das Element, das zuletzt in die Tabelle eingetragen wurde. In eine volle Tabelle können keine weiteren Eintragungen gemacht werden. Der Wert von LEN muß zwischen 1 und 256 liegen.

Nach erfolgreicher Ausführung gibt die Funktion den Stromfluß nach rechts weiter.



Parameter:

Parameter	Beschreibung
enable	Die LIFOWRT-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
TB	TB enthält die Elemente im Stack.
PTR	PTR wird erhöht und zeigt dann auf das nächste LIFO-Element, das geschrieben werden soll.
IN	IN enthält das Element, das in den Stack eingetragen werden soll.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben und PTR < LEN ist .
FL	Der Ausgang FL wird durchgeschaltet, wenn das letzte Element im Stack eingetragen wurde.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB		o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
IN	•	o	o	o	o	Δ†	o	•	•	•	•	•	•	
ok	•													•
FL	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

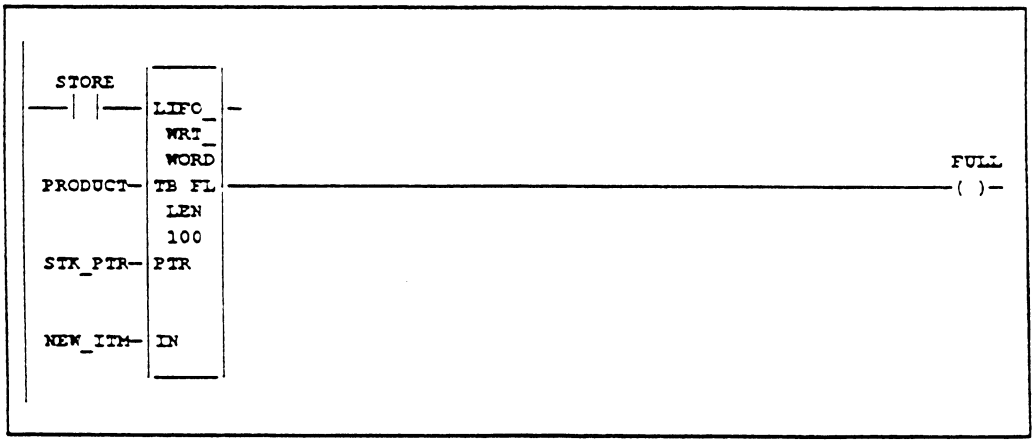
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.

Δ = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im folgenden Beispiel ist PRODUCT ein LIFO-Stack mit 100 wortstrukturierten Elementen. Ist der Freigabeeingang STORE wahr, dann wird ein Datenelement von NEW_ITEM oben im Stack eingetragen. Die mit STK_PTR gekennzeichnete Referenz enthält den Tabellenpointer. Die Ausgangsspule FULL zeigt an, daß der Stack voll ist.

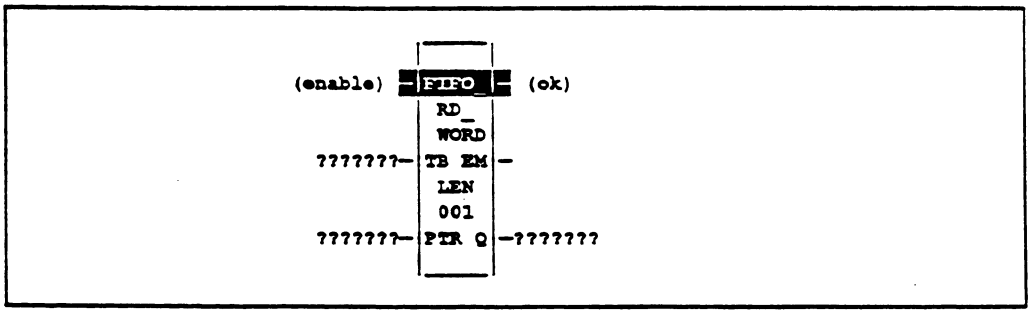


FIFORD (INT, DINT, UINT, WORD, DWORD)

Mit der FIFORD-Funktion können Sie Daten aus einer als FIFO-Stack organisierten Tabelle auslesen. Die FIFORD-Funktion entfernt den Eintrag an der Pointerposition und dekrementiert den Pointer um 1. Wird der unterste Eintrag aus der Tabelle entfernt, dann werden alle anderen Einträge um eins nach unten verschoben und der Pointer um 1 erniedrigt. Die FIFO-Lesefunktion wird zusammen mit der FIFOWRT-Funktion verwendet, die den Pointer erhöht und Daten am Stackanfang einträgt.

Die FIFO-Lesefunktion besitzt drei Eingangs- und drei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann werden die Daten aus der ersten Tabellenposition zum Ausgang Q kopiert. Danach werden alle Einträge in der Tabelle nach unten zur nächstniedrigen Position verschoben. Abschließend wird der Pointer dekrementiert. Wird hierdurch die Pointerposition 0 erreicht, dann wird Ausgang EM durchgeschaltet. Dieser Ausgang zeigt somit an, ob die Tabelle leer ist. Der Wert von LEN muß zwischen 1 und 256 liegen.

Die Funktion gibt Stromfluß nach rechts weiter, wenn der Pointerwert größer als Null und kleiner als der für LEN angegebene Wert ist.



Parameter:

Parameter	Beschreibung
enable	Die FIFORD-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist. Im FIFO-Stack wird immer das erste Element gelesen.
TB	TB enthält die Elemente im Stack.
PTR	PTR zeigt auf das letzte Element im FIFO-Stack.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben ist und 0 < PTR ≤ LEN.
EM	Der Ausgang EM wird durchgeschaltet, wenn das letzte Element im FIFO-Stack gelesen wurde.
Q	Der Ausgang Q enthält das aus dem Stack gelesene Element.

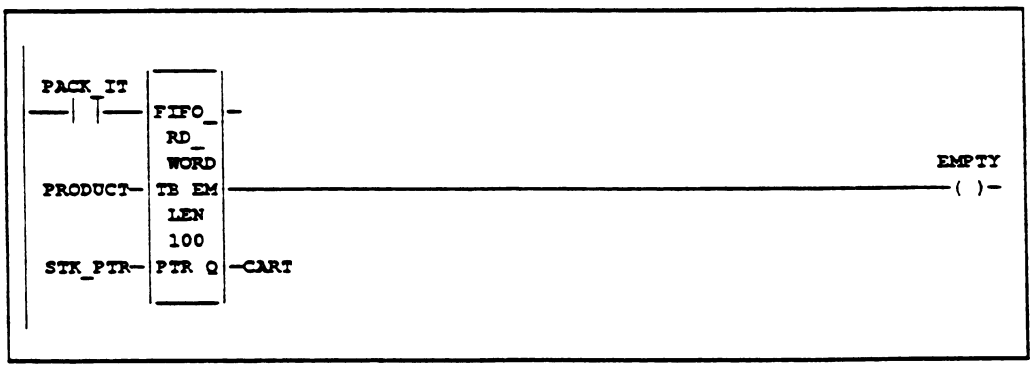
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB	•	o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
ok	•													•
EM	•													•
Q	•	o	o	o	o	Δ†	o	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.
Δ = Datentyp nur für WORD-Daten zulässig.
† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

Im folgenden Beispiel ist PRODUCT ein FIFO-Stack mit 100 wortstrukturierten Elementen. Ist der Freigabeeingang PACK_IT wahr, dann wird das unterste Datenelement aus dem Stack in die durch die symbolische Adresse CART gekennzeichnete Referenz kopiert. Die mit STK_PTR gekennzeichnete Referenz enthält den Tabellenpointer. Die Ausgangsspule EMPTY zeigt an, daß der Stack leer ist.

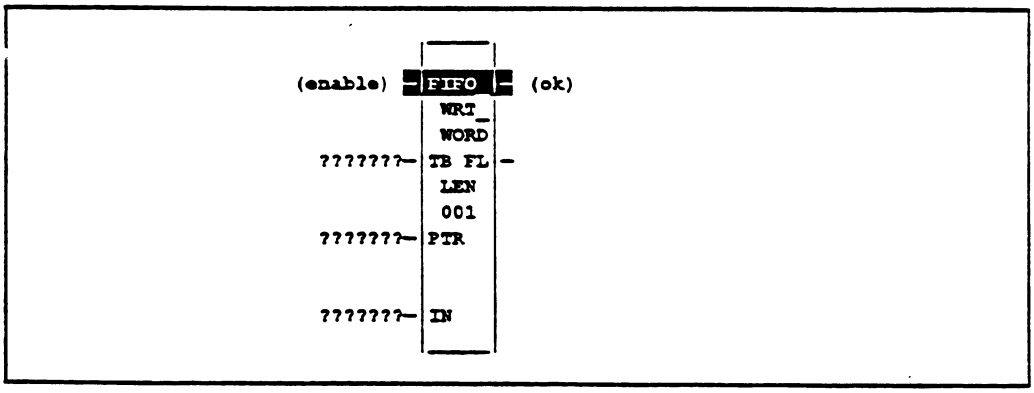


FIFOWRT (INT, DINT, UINT, WORD, DWORD)

Mit der FIFOWRT-Funktion können Sie Daten zu einer als FIFO-Stack organisierten Tabelle hinzufügen. Die FIFO-Schreibfunktion wird zusammen mit der FIFORD-Funktion verwendet, die Einträge aus dem Stack wegnimmt und den Pointer erniedrigt.

Die FIFO-Schreibfunktion besitzt vier Eingangs- und zwei Ausgangsparameter. Empfängt die Funktion Stromfluß, dann wird der Pointer um 1 erhöht und die neuen Daten werden an der Pointerposition eingetragen. Stand der Pointer bereits auf der letzten Tabellenposition, dann werden keine Daten eingetragen und die Funktion schaltet den Stromfluß nicht nach rechts durch. Der Pointer zeigt immer auf das Element, das zuletzt in die Tabelle eingetragen wurde. In eine volle Tabelle können keine weiteren Eintragungen gemacht werden. Der Wert von LEN muß zwischen 1 und 256 liegen.

Nach erfolgreicher Ausführung gibt die Funktion den Stromfluß nach rechts weiter.



Parameter:

Parameter	Beschreibung
enable	Die FIFOWRT-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
TB	TB enthält die Elemente im FIFO-Stack.
PTR	PTR zeigt auf das letzte LIFO-Element im FIFO-Stack.
IN	IN enthält das Element, das in den FIFO-Stack eingetragen werden soll.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion freigegeben und PTR < LEN ist .
FL	Der Ausgang FL wird durchgeschaltet, wenn das letzte Element im FIFO-Stack eingetragen wurde.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
TB		o	o	o	o	Δ†	o	•	•	•	•	•		
PTR		•	•	•	•		•	•	•	•	•	•		
IN	•	o	o	o	o	Δ†	o	•	•	•	•	•	•	
ok	•													•
FL	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

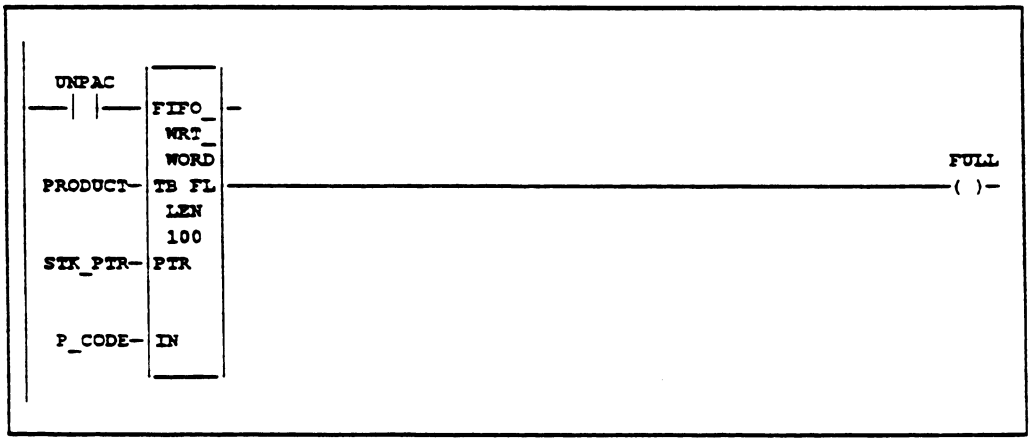
o = Datentyp nur für UINT, INT oder WORD-Daten zulässig.

Δ = Datentyp nur für WORD-Daten zulässig.

† = Nur %SA, %SB oder %SC. %S kann nicht verwendet werden.

Beispiel:

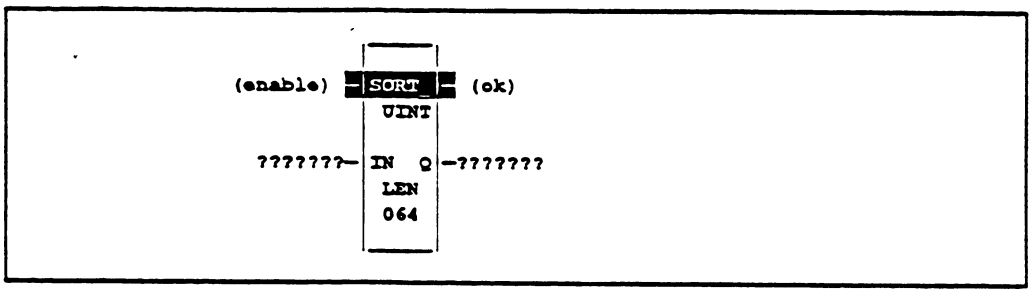
Im folgenden Beispiel ist PRODUCT ein LIFO-Stack mit 100 wortstrukturierten Elementen. Ist der Freigabeeingang UNPACK wahr, dann wird ein Datenelement von P_CODE oben im Stack eingetragen. Die mit STK_PTR gekennzeichnete Referenz enthält den Tabellenpointer. Die Ausgangsspule FULL zeigt an, daß der Stack voll ist.



SORT (INT, UINT, WORD)

Mit der SORT-Funktion können Sie ein Feld in aufsteigender Reihenfolge sortieren. Die Funktion besitzt zwei Eingangs- und zwei Ausgangsparameter. EN ist ein Boolscher Freigabeeingang, IN das zu sortierende Feld. OK ist ein Boolscher Ausgang, der Stromfluß weitergibt; Q ist ein Feld mit ganzen Zahlen, das die Indizes angibt, die die sortierten Elemente in ihrem ursprünglichen Feld oder ihrer Liste hatten. Q, das genau die gleiche Größe wie IN besitzt, enthält auch eine Angabe (LEN) über die Anzahl der zu sortierenden Elemente.

Die Sortierfunktion kann nur auf Felder mit maximal 64 Elementen angewandt werden. Ist EN wahr, dann werden alle Elemente von IN in aufsteigender Reihenfolge auf der Basis ihrer Typen sortiert. Gleichzeitig wird das Feld Q angelegt, das die ursprüngliche Position der einzelnen sortierten Elemente angibt. OK wird immer wahr.



Parameter:

Parameter	Beschreibung
enable	Die SORT-Funktion wird ausgeführt, wenn dieser Eingang aktiviert ist.
IN	IN enthält das Feld, dessen Elemente sortiert werden sollen. Am Ende des Sortiervorgangs sind die Elemente von IN sortiert.
ok	Dieser Ausgang wird immer dann durchgeschaltet, wenn LEN kleiner als 65 ist. Der Parameter LEN wird jedoch nicht auf 64 begrenzt. Ist LEN größer als 64, dann bearbeitet der Funktionsblock lediglich die ersten 64 Einheiten von LEN.
Q	Ausgang Q enthält ein Indexfeld, das die Positionen der sortierten Elemente vor dem Sortiervorgang angibt.

Zulässige Speichertypen:

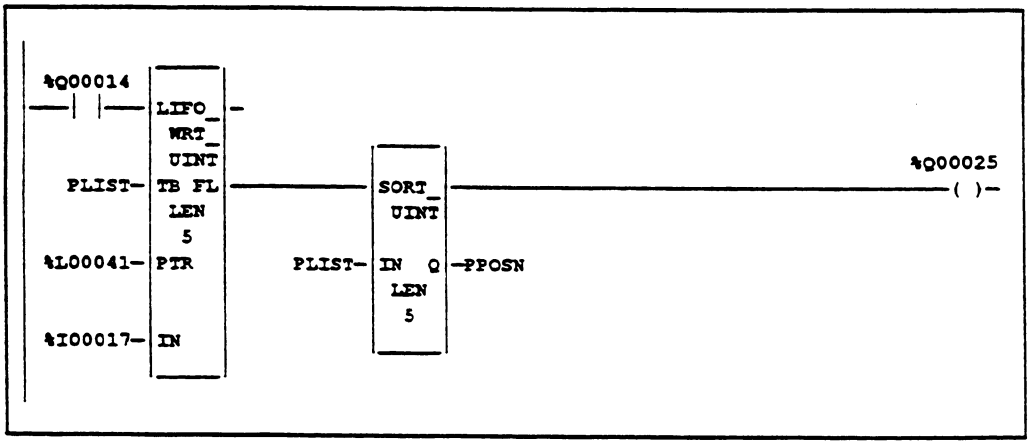
Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN		•	•	•	•		•	•	•	•	•	•		
ok	•													•
Q	•	•	•	•	•		•	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Im folgenden Beispiel werden jedesmal, wenn %Q00014 wahr ist, neue Teilenummern in das Teilefeld PLIST eingeschoben. Ist das Feld voll, dann wird es sortiert und der Ausgang %Q00025 durchgeschaltet. Das Feld PPOSN enthält die ursprüngliche Anordnung der nun sortierten Elemente.

Ist PLIST z.B. eine Feld mit fünf Elementen, die vor dem Sortiervorgang die Werte 25, 67, 12, 35, 14 enthielten, dann enthalten diese Elemente nach dem Sortieren die Werte 12, 14, 25, 35, 67. In PPOSN stehen dann die Werte 3, 5, 1, 4, 2.



KAPITEL 10

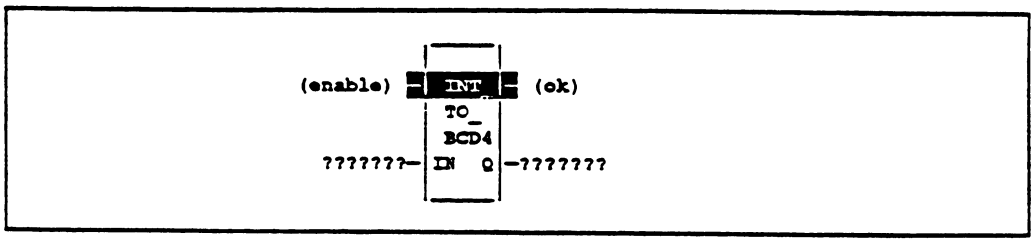
Mit den Konvertierungsfunktionen können Sie das Format eines Datenelements umwandeln. Zahlreiche Programmanweisungen, wie z.B. die arithmetischen Funktionen, verlangen die Verwendung eines bestimmten Datentyps. In diesem Kapitel werden die folgenden Konvertierungsfunktionen beschrieben:

Funktion	Beschreibung
BCD-4	Diese Funktion konvertiert eine ganze Zahl mit oder ohne Vorzeichen in 4-stelliges BCD-Format.
BCD-8	Diese Funktion konvertiert eine doppeltgenaue ganze Zahl in 4-stelliges BCD-Format.
UINT	Diese Funktion konvertiert eine Zahl im BCD-4-Format, eine ganze Zahl mit Vorzeichen oder eine doppeltgenaue ganze Zahl in eine ganze Zahl ohne Vorzeichen.
INT	Diese Funktion konvertiert eine Zahl im BCD-4-Format oder eine ganze Zahl ohne Vorzeichen oder mit doppelter Genauigkeit in eine vorzeichenbehaftete ganze Zahl.
DINT	Diese Funktion konvertiert eine Zahl im BCD-8-Format oder eine ganze Zahl mit oder ohne Vorzeichen in eine doppeltgenaue ganze Zahl.
REAL	Diese Funktion konvertiert eine Zahl im BCD-4 oder BCD-8-Format oder eine ganze Zahl mit oder ohne Vorzeichen oder eine doppeltgenaue ganze Zahl in eine reelle Zahl.
TRUN	Diese Funktion rundet eine reelle Zahl nach unten ab.

Die .i.BCD-4-Funktion gibt das vierstellige BCD-Äquivalent einer ganzen Zahl mit oder ohne Vorzeichen aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Die Konvertierung in BCD-Format ist zum Beispiel erforderlich für die Ansteuerung von BCD-codierten LED-Anzeigen oder für die Einstellung externer Geräte (z.B. schnelles Zählmodul).

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Solange das Ergebnis der Konvertierung im Bereich zwischen 0 und 9999 liegt, schaltet die Funktion den Stromfluß durch, wenn sie aktiviert wird.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den ganzzahligen Wert, der in BCD-4 konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde.
Q	Ausgang Q enthält das BCD-4-Aquivalent des in IN spezifizierten Wertes.

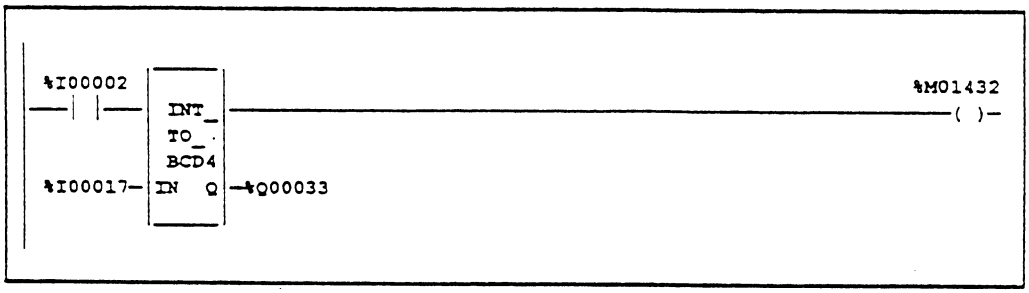
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	•	•	•	•		•	•	•	•	•	•	•	
ok	•													•
Q	•	•	•	•	•		•	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

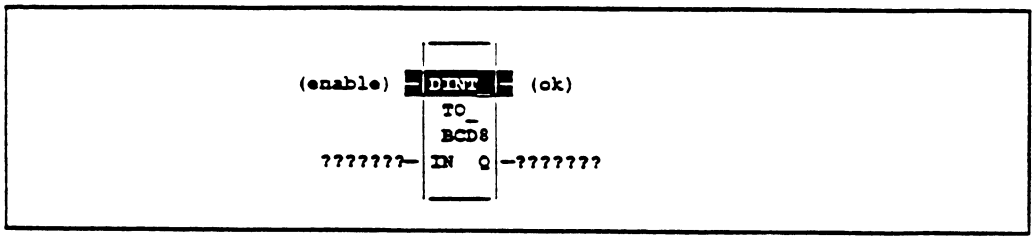
Beispiel:

Wird im folgenden Beispiel der Eingang %I00002 gesetzt und liegt kein Fehler vor, dann wird die an den Eingängen %I00017 bis %I00032 anliegende ganze Zahl in vier BCD-Stellen umgewandelt. Das Ergebnis wird in den Speicheradressen %Q00033 bis %Q00048 abgelegt. Mit der Spule %M01432 kann überprüft werden, ob die Konvertierung erfolgreich war.



Die i.BCD-8-Funktion gibt das achtstellige BCD-Äquivalent einer doppeltgenauen ganzen Zahl aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Solange das Ergebnis der Konvertierung im Bereich zwischen 0 und 99999999 liegt, schaltet die Funktion den Stromfluß durch, wenn sie aktiviert wird.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den ganzzahligen Wert, der in BCD-4 konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde.
Q	Ausgang Q enthält das BCD-4-Aquivalent des in IN spezifizierten Wertes.

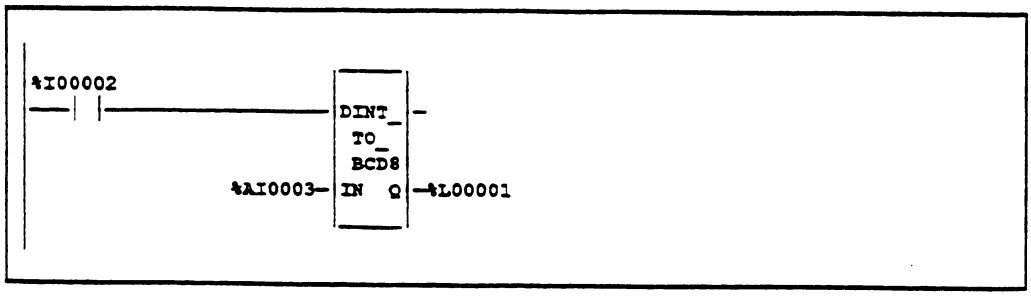
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•							•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

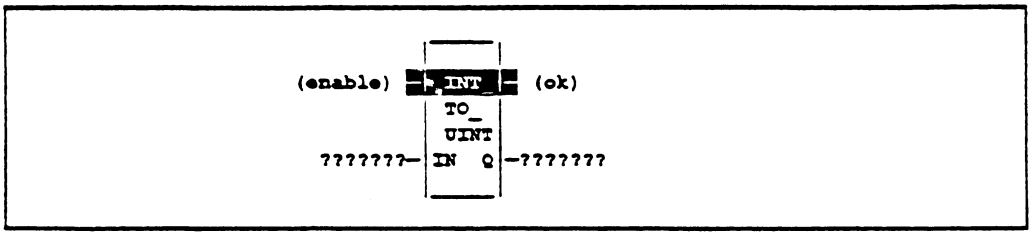
Wird im folgenden Beispiel der Eingang %I00002 gesetzt und liegt kein Fehler vor, dann wird die am Eingang %AI0003 anliegende doppeltgenaue ganze Zahl in acht BCD-Stellen umgewandelt. Das Ergebnis wird in den Speicheradressen %L00001 bis %L00002 abgelegt.



Die .i.UINT-Funktion gibt das ganzzahlige Äquivalent von ganzen Zahlen mit Vorzeichen, doppeltgenauen ganzen Zahlen, BCD-4-Daten oder reellen Werten aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Mit der agegeber) bilden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Solange das Ergebnis der Konvertierung im Bereich zwischen 0 und +65.535 liegt, schaltet die Funktion den Stromfluß durch, wenn sie aktiviert wird.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den Wert, der in eine ganze Zahl ohne Vorzeichen konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde.
Q	Ausgang Q enthält das ganzzahlige Aquivalent ohne Vorzeichen des in IN spezifizierten Wertes.

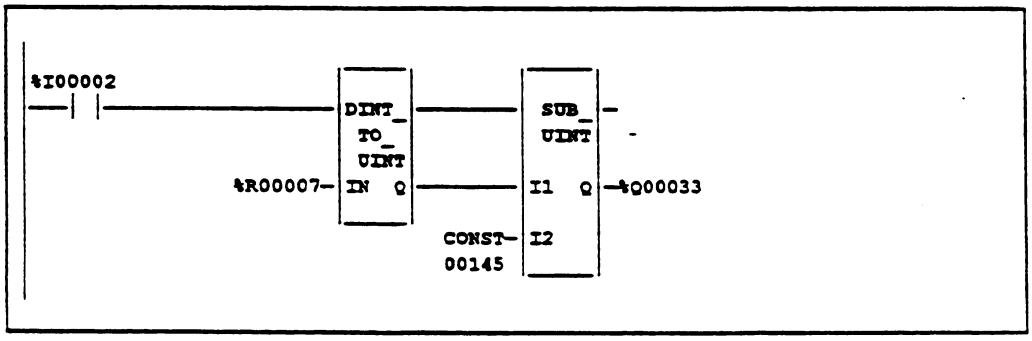
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	•	•	•	•		•	•	•	•	•	•		

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Nicht zulässig für DINT_TO_UINT

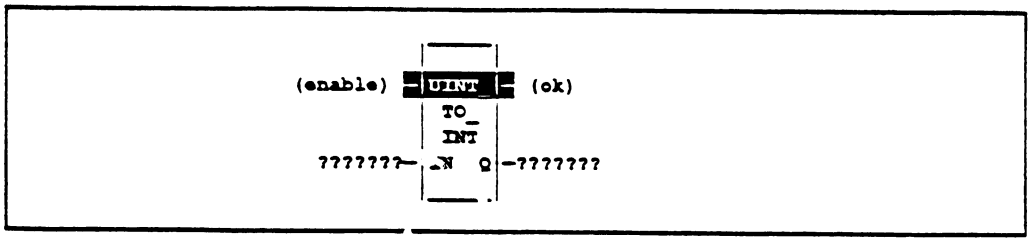
Beispiel:

Wird im folgenden Beispiel der Eingang %I00002 gesetzt und liegt kein Fehler vor, dann wird die am Eingang %R00007 anliegende doppeltgenaue ganze Zahl in eine vorzeichenlose ganze Zahl umgewandelt und an die SUB-Funktion weitergegeben, in der von dieser Zahl die Konstante 145 subtrahiert wird.. Das Ergebnis wird in der Ausgangsreferenz %Q00033 abgelegt.



Die .i.INT-Funktion gibt das ganzzahlige Äquivalent einer ganzen Zahl ohne Vorzeichen, einer doppeltgenauen ganzen Zahl oder eines 4-stelligen BCD-Wertes aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Die Funktion schaltet den Stromfluß immer durch, wenn sie aktiviert wird und die Daten innerhalb des zulässigen Bereichs liegen.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den Wert, der in einen ganzzahligen Wert konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn 'enable' durchgeschaltet wird und die Daten innerhalb des zulässigen Bereichs liegen.
Q	Ausgang Q enthält die ganzzahlige Form des in IN spezifizierten Wertes.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•	•	•	•	•		•	•	•	•	•	•		

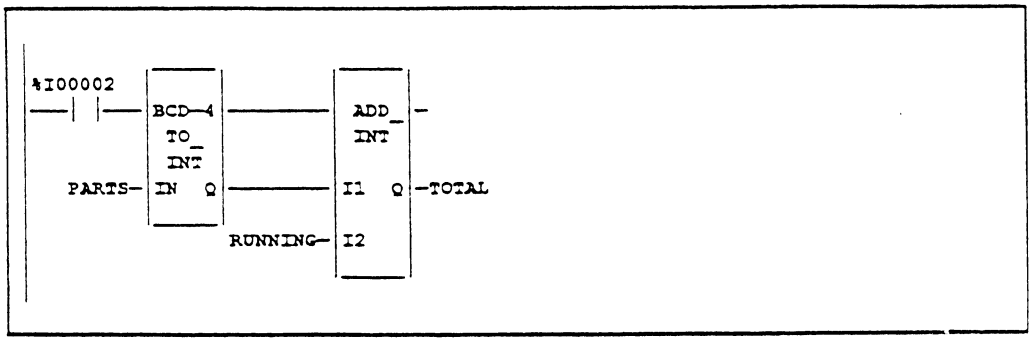
Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Nicht zulässig für DINT_TO_INT

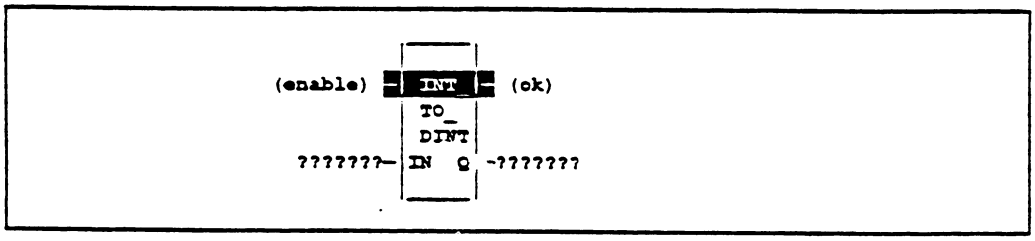
Beispiel:

Wird im folgenden Beispiel der Eingang %I00002 gesetzt, dann wird der vierstellige BCD-Wert in PARTS in eine vorzeichenbehaftete ganze Zahl konvertiert, die an die ADD-Funktion weitergegeben wird. Hier wird der Wert zu der durch die Referenz RUNNING spezifizierten vorzeichenbehafteten ganzen Zahl addiert. Das Ergebnis wird von der ADD-Funktion an die Referenz TOTAL ausgegeben.



Die .i.DINT-Funktion gibt das ganzzahlige Äquivalent doppelter Genauigkeit einer ganzen Zahl mit oder ohne Vorzeichen oder eines 8-stelligen BCD-Wertes aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Die Funktion schaltet den Stromfluß immer durch, wenn sie aktiviert wird und der reelle Wert innerhalb des zulässigen Bereichs liegt.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den Wert, der in einen ganzzahligen Wert doppelter Genauigkeit konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn 'enable' durchgeschaltet wird und der reelle Wert innerhalb des zulässigen Bereichs liegt.
Q	Ausgang Q enthält die doppeltgenaue ganzzahlige Form des in IN spezifizierten Wertes.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•		

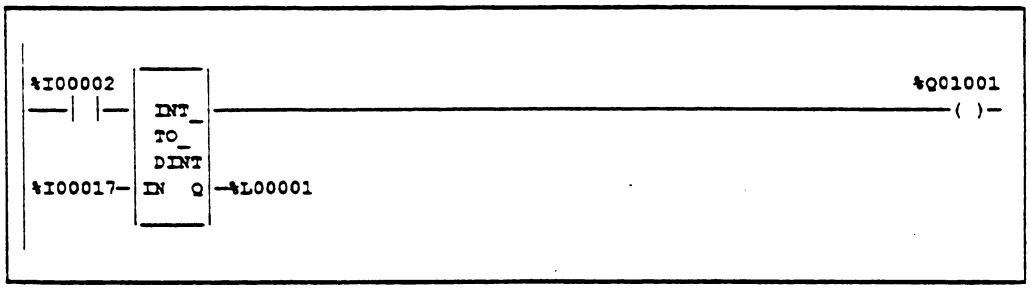
Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Nicht zulässig für BCD8_TO_DINT

Beispiel:

Wird im folgenden Beispiel der Eingang %I00002 gesetzt, dann wird der ganzzahlige Wert am Eingang %I00017 in eine doppeltgenaue ganze Zahl konvertiert. Das Ergebnis wird in die Adresse %L00001 eingetragen. Ausgang %Q01001 wird durchgeschaltet, wenn die Funktion erfolgreich ausgeführt wurde.



GFK-0265D-GE

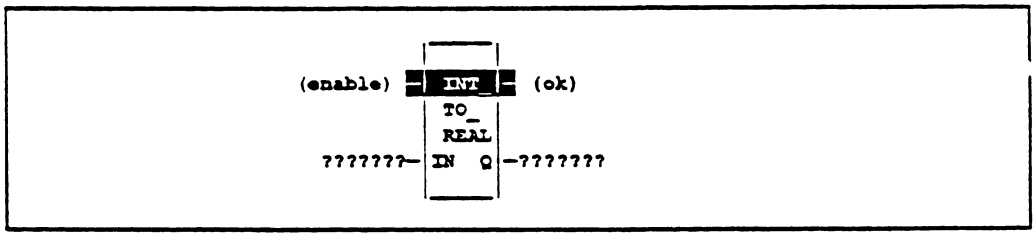
→REAL (INT, UINT, DINT, BCD-4, BCD-8)

Die REAL-Funktion gibt den reellen Wert der Eingangsdaten aus. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Die Funktion schaltet den Stromfluß immer durch, wenn sie aktiviert wird und der ausgegebene Wert innerhalb des zulässigen Bereichs liegt.

HINWEIS

Bei der Konvertierung von DINT oder BCD-8 in REAL kann Genauigkeit verlorengehen, da die Anzahl signifikanter Bits auf 24 reduziert wird.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den ganzzahligen Wert, der in einen reellen Wert konvertiert werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei durchgeführt wird.
Q	Ausgang Q enthält die reelle Form des in IN spezifizierten Wertes.

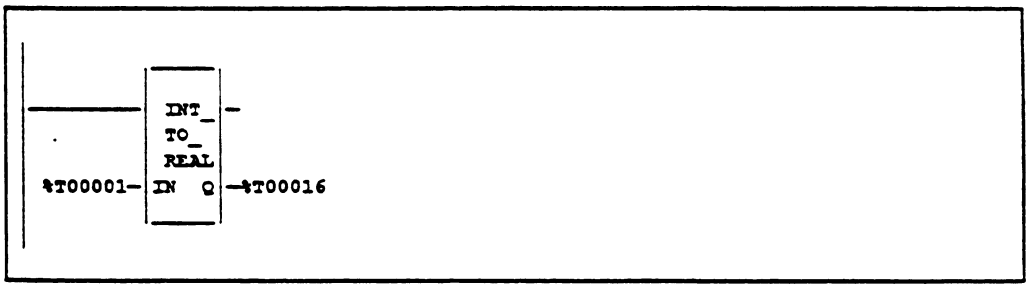
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•	o	o	o	o		o	•	•	•	•	•	•	
ok	•													•
Q	•							•	•	•	•	•	•	

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.
o = Nicht zulässig für DINT_TO_REAL oder BCD8_TO_REAL.

Beispiel:

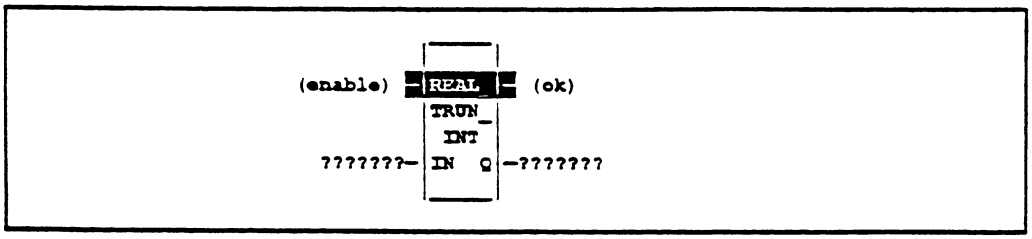
Im folgenden Beispiel ist 678 der ganzzahlige Wert von IN. 678.000 wird als Ergebnis in %T00016 eingetragen.



TRUN (INT, DINT)

Mit der TRUN-Funktion werden reelle Zahlen nach unten abgerundet. Die ursprünglichen Daten werden dabei nicht verändert. Die Ausgangsdaten können direkt als Eingabe zu einer anderen Programmfunktion verwendet werden.

Erhält die Funktion Stromfluß, dann führt sie die Konvertierung durch und stellt das Ergebnis am Ausgang Q bereit. Die Funktion schaltet den Stromfluß immer durch, wenn sie aktiviert wird, der ausgegebene Wert innerhalb des zulässigen Bereichs liegt, und IN eine Zahl ist.



Parameter:

Parameter	Beschreibung
enable	Die Konvertierung wird durchgeführt, wenn die Funktion freigegeben ist.
IN	IN enthält eine Referenz für den reellen Wert, der abgerundet werden soll.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei durchgeführt wird, der Wert innerhalb des zulässigen Bereichs liegt, und IN eine Zahl ist.
Q	Ausgang Q enthält den abgerundeten INT- oer DINT-Wert des in IN spezifizierten Wertes.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
IN	•							•	•	•	•	•	•	
ok	•													•
Q	•	o	o	o	o		o	•	•	•	•	•		

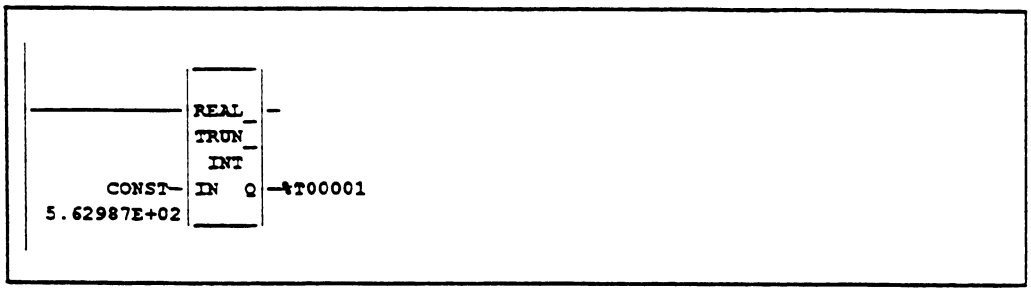
Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

o = Nur zulässig für REAL_TRUN_INT.

Beispiel:

Im folgenden Beispiel wird die angezeigte Konstante abgerundet und das ganzzahlige Ergebnis 562 in %T00001 eingetragen.



KAPITEL 11

In diesem Kapitel werden die Steuerfunktionen beschrieben, mit denen die Programmausführung eingeschränkt und der von der CPU verfolgte Weg der Programmausführung abgeändert werden kann.

Funktion	Beschreibung
CALL	Leitet die Programmausführung zu einem angegebenen Unterprogramm.
DOIO	Aktualisiert während eines Zyklus unmittelbar einen bestimmten Bereich von Ein- oder Ausgängen. (Enthält eine DO I/O-Funktion Referenzadressen eines Moduls, dann werden sämtliche Ein- oder Ausgänge dieses Moduls aktualisiert, teilweise Aktualisierungen eines Moduls werden nicht durchgeführt). Wahlweise kann eine Kopie der bearbeiteten E/A im internen Speicher und nicht bei den echten Eingangspunkten abgelegt werden.
SUSIO	Hebt für die Dauer eines Zyklus die normale E/A-Aktualisierung auf; nur die durch die DOIO-Anweisungen bestimmten E/A-Punkte werden aktualisiert.
MCR	Programmierung einer Hauptsteuerrelaisfunktion. Die MCR-Funktion veranlaßt, daß alle Strompfade zwischen MCR und dem nächsten ENDMCR ohne Stromfluß ausgeführt werden.
ENDMCR	Zeigt an, daß für die Ausführung der nachfolgenden Logik wieder Stromfluß erforderlich ist.
JUMP	Bewirkt, daß bei der Programmausführung zu einer angegebenen Stelle (durch eine Marke gekennzeichnet; siehe LABEL) im Programm gesprungen wird.
LABEL	Gibt das Ziel einer JUMP-Anweisung an.
COMMNT	Legt einen Kommentar (Strompfadkommentar) im Programm an. Nachdem die Anweisung programmiert wurde, kann der Text durch "Zoomen" der Anweisung eingegeben werden.
SVCREQ	Anforderung einer der folgenden SPS-Spezialdienste: Zeitglied für konstante Zyklusdauer lesen/stellen Windowwerte lesen Zustand und Werte im Programmiergeräte-Kommunikationswindow ändern Zustand und Werte im System-Kommunikationswindow ändern Echtzeituhr lesen/stellen Zeitüberwachung (Watchdog) rücksetzen Zykluszeit seit Zyklusbeginn lesen Programmnamen des aktuellen Blocks lesen SPS-Kennung lesen SPS-Betriebszustand lesen SPS abschalten Fehlertabellen löschen Letzten Fehlertabelleneintrag lesen Betriebszeituhr lesen E/A-Interrupt maskieren/entmaskieren E/A-Override-Zustand lesen RUN freigeben/sperren Fehlertabellen lesen
PID	Standard-ISA PID-Algorithmus (PIDISA) Unabhängiger PID-Algorithmus (PIDIND)

CALL

Mit der CALL-Funktion können Sie die Programmausführung auf einen bestimmten Programmblock umlenken. Eine CALL-Funktion kann in jedem Hauptprogrammblock oder Programmblock verwendet werden.

HINWEIS

Maximal können in einem Programmblock 64 Programmblock-Aufrufe auftreten.

Erhält die CALL-Funktion Stromfluß, dann leitet sie die Zyklusbearbeitung direkt auf den angegebenen Programmblock, der dann abgearbeitet wird. Nachdem der Programmblock bearbeitet wurde, geht der Programmablauf wieder zu dem Punkt in der Logik zurück, der unmittelbar auf die CALL-Anweisung folgt.

DOIO

Mit der DO I/O-Funktion können Ein- und Ausgänge bei laufendem Programm für einen Zyklus aktualisiert werden. DO I/O kann zusammen mit der Funktion SUSPEND I/O verwendet werden, die den normalen E/A-Zyklus anhält. Die Funktion kann auch dazu benutzt werden, um bestimmte E/A zusätzlich zum normalen E/A-Zyklus während des Programms zu aktualisieren.

Wenn Eingangsreferenzen angegeben werden, dann können durch die Funktion DO I/O die neuesten Eingangswerte für das Programm erfaßt werden. Werden Ausgangsreferenzen angegeben, dann aktualisiert DO I/O die Ausgänge auf der Grundlage der neuesten im E/A-Speicher abgelegten Werte. Bei der E/A werden immer komplette E/A-Module aktualisiert, falls erforderlich, berichtigt die SPS während der Funktionsausführung die Referenzen.

HINWEIS

Die DO I/O-Funktion wird nur für Serie 90-70 E/A-Module unterstützt, nicht für Genius-E/A-Module.

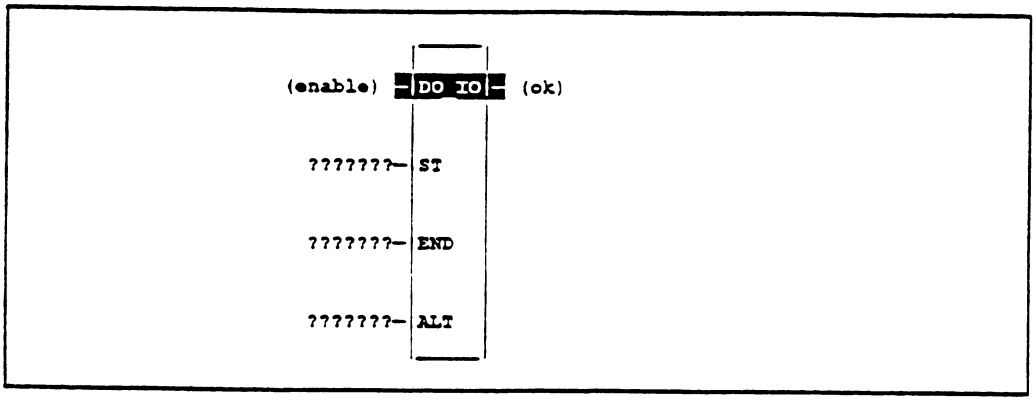
Die DO I/O-Funktion besitzt vier Eingangs- und einen Ausgangsparameter. Erhält die Funktion Stromfluß und sind Eingangsreferenzen angegeben, dann werden die E/A-Punkte zwischen der Anfangsreferenz ST und END abgefragt. Wurde für ALT eine Referenz angegeben, dann wird eine Kopie der neuen Eingangswerte ab dieser Referenz im Speicher abgelegt und die realen Eingangspunkte werden nicht aktualisiert. ALT muß genauso groß sein wie der abgefragte Referenztyp. Werden für ST und END diskrete Referenzen verwendet, dann muß ALT auch diskret sein. Wird für ALT keine Referenz angegeben, dann werden die realen Eingangspunkte aktualisiert.

Erhält die DO I/O-Funktion Stromfluß und sind Ausgangsreferenzen angegeben, dann werden die Ausgangspunkte zwischen der Anfangsadresse ST und der Endeadresse END in die Ausgangsmodule geschrieben. Sollen von %Q oder %AQ verschiedene Ausgangswerte aus dem internen Speicher in die Ausgangsmodule geschrieben werden, dann kann die Anfangsreferenz für ALT angegeben werden. Der Bereich der zu den Ausgangsmodulen übertragenen Ausgangswerte kann durch die Anfangsadresse ST und die Endeadresse END angegeben werden.

Die Funktion wird solange ausgeführt, bis entweder alle Eingänge des gewählten Bereiches abgefragt wurden oder alle Ausgänge auf den E/A-Modulen aktualisiert wurden. Die Programmausführung kehrt dann zur nächsten Funktion nach der DO I/O-Funktion zurück.

Die Funktion gibt Stromfluß nach rechts weiter, wenn sie Stromfluß erhält, mit Ausnahme von:

- Nicht alle Referenzen des angegebenen Typs sind im gewählten Bereich vorhanden.
- Ein oder mehrere Buscontroller haben nicht rechtzeitig geantwortet oder die Anforderung zur Datenübertragung zurückgewiesen.
- Die CPU kann die von der Funktion angelegte temporäre E/A-Liste nicht richtig verarbeiten.
- Die SPS kann nicht zum Systembus schreiben oder von dort lesen. Hierdurch entsteht eine Meldung "Systembusfehler".
- Der angegebene Bereich enthält E/A-Module, die mit einem Fehler "E/A-Verlust" verknüpft sind.



Parameter:

Parameter	Beschreibung
enable	Wird die Funktion freigegeben, dann wird ein begrenzter Ein- oder Ausgabezyklus durchgeführt.
ST	ST ist die Anfangsadresse eines Satzes von Ein- oder Ausgangspunkten oder Worten, die bedient werden sollen.
END	END ist die Endeadresse eines Satzes von Ein- oder Ausgangspunkten oder Worten, die bedient werden sollen.
ALT	Beim Eingabezyklus gibt ALT die Adresse an, in der die abgefragten Eingangspunkte/Worte gespeichert werden sollen. Beim Ausgabezyklus gibt ALT die Adresse an, von der die Ausgangspunkte/Worte abgeholt werden können, die zu den E/A-Modulen geschickt werden.
OK	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion normal ausgeführt wird.

Zulässige Speichertypen:

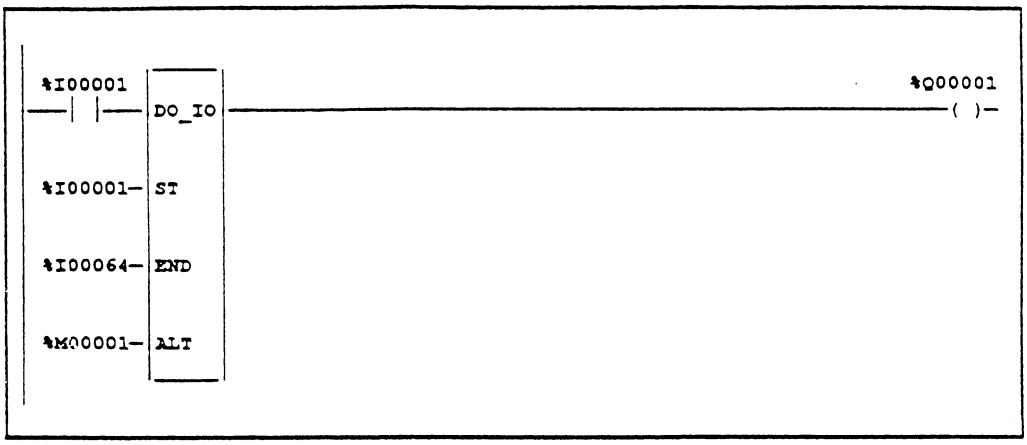
Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
ST		•	•								•	•		
END		•	•								•	•		
ALT		•	•	•	•		•	•						•
ok	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

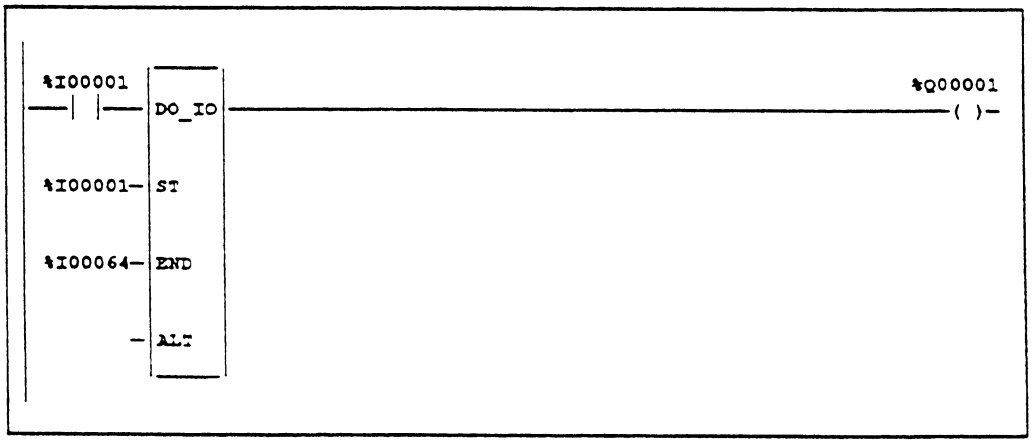
Eingabebeispiel 1:

Wird im folgenden Beispiel der Freigabeeingang %I00001 "wahr", dann werden die Referenzen %I00001 bis %I00064 abgefragt und %Q00001 wird durchgeschaltet. Eine Kopie der abgefragten Eingänge wird im internen Speicher ab Referenz %M00001 bis Referenz %M00064 abgelegt. Die echten Eingangspunkte werden nicht aktualisiert. Mit dieser Funktionsart können die aktuellen Werte der Eingangspunkte mit den Werten der Eingangspunkte am Anfang des Zyklus verglichen werden.



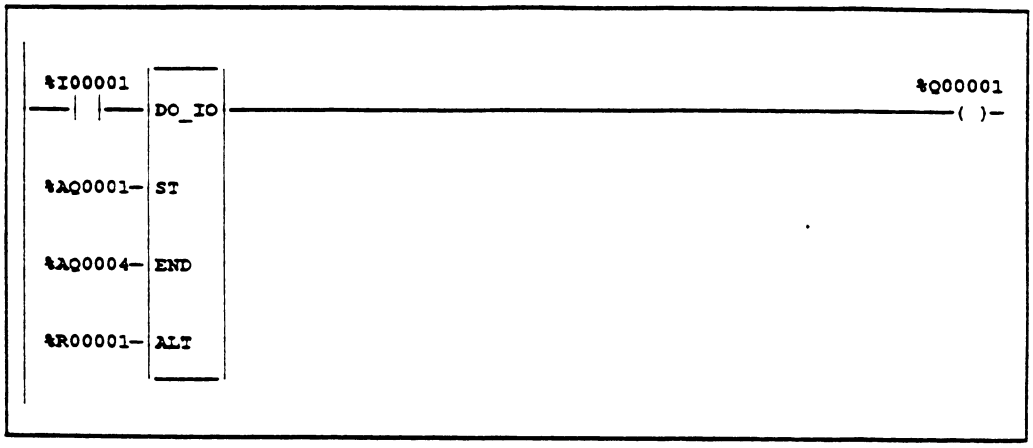
Eingabebeispiel 2:

Wird im folgenden Beispiel der Freigabeeingang %I00001 "wahr", dann werden die Referenzen %I00001 bis %I00064 abgefragt und %Q00001 wird durchgeschaltet. Die abgefragten Eingangswerte werden im Eingangs-Zustandsspeicher in den Referenzen %I00001 bis %I00064 abgelegt. Mit dieser Art der Funktion können Eingangspunkte einmal oder mehrmals während des Programmausführungsteils des Zyklus abgefragt werden.



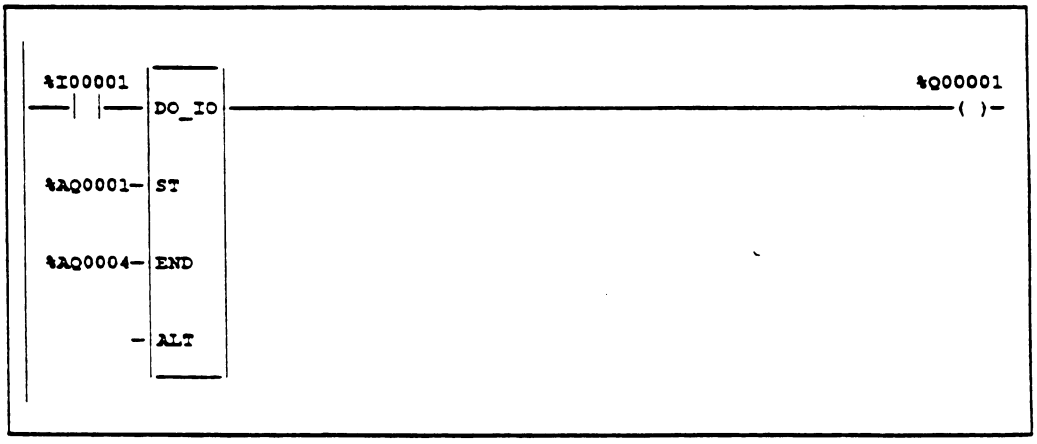
Ausgabebeispiel 1:

Ist im folgenden Beispiel der Freigabeeingang %I00001 "wahr", dann werden die Werte aus den Referenzen %R00001 bis %R00004 in die Analogkanäle %AQ0001 bis %AQ0004 geschrieben und %Q00001 wird durchgeschaltet. Die Werte aus %AQ0001 bis %AQ0004 werden nicht zu den Analog-Ausgangsmodulen übertragen.



Ausgabebeispiel 2:

Ist im folgenden Beispiel der Freigabeeingang %I00001 "wahr", dann werden die Werte aus den Referenzen %AQ0001 bis %AQ0004 zu den Analog-Ausgangskanälen AQ0001 bis %AQ0004 geschrieben und %Q00001 wird durchgeschaltet.



SUSIO

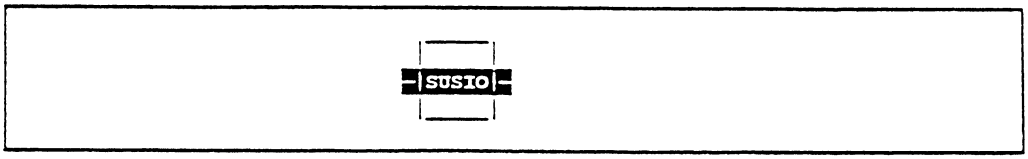
Mit der SUSPEND I/O-Funktion kann die normale E/A-Aktualisierung für die Dauer eines CPU-Zyklus angehalten werden. Während des nächsten Ausgabezyklus werden alle Ausgänge auf ihren momentanen Zuständen festgehalten. Während des nächsten Eingabezyklus werden die Eingangsreferenzen nicht mit den Daten der Eingänge aktualisiert. Die CPU überprüft jedoch während des Eingabezyklus, ob die Genius-Buscontroller ihre vorherige Ausgangsaktualisierung beendet haben.

HINWEIS

Die SUSPEND I/O-Funktion hält die gesamte E/A an (analog und diskret, integrierte E/A oder Genius-E/A).

Die SUSPEND I/O-Funktion besitzt einen Eingang und einen Ausgang. Erhält die Funktion Stromfluß, dann wird die gesamte E/A-Aktualisierung mit Ausnahme der durch die DO I/O-Funktion gesteuerten E/A angehalten. Würde die Funktion an der linken Stromschiene im Kontaktplanprogramm ohne Freigabelogik angebracht, dann würde kein regulärer E/A-Zyklus mehr durchgeführt.

Die SUSPEND I/O-Funktion schaltet den Stromfluß immer durch, wenn sie Stromfluß empfängt.



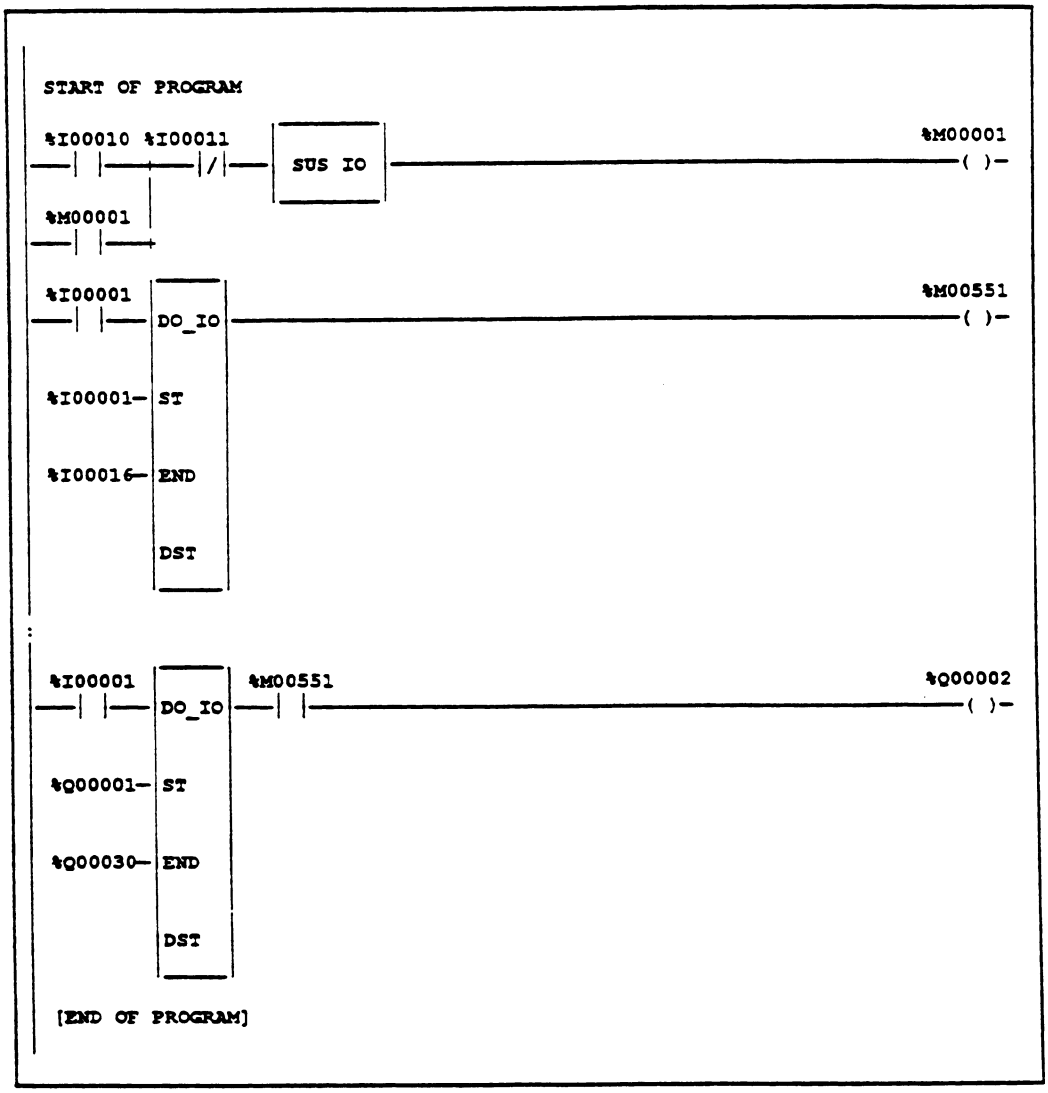
Beispiel:

Das nachstehende Beispiel zeigt, wie mit SUS I/O und DO I/O der E/A-Zyklus angehalten und dann die Aktualisierung für bestimmte E/A-Punkte freigegeben wird.

Die Eingänge %I00010 und %I00011 bilden mit dem Kontakt von %M00001 einen Haltekreis, der SUS I/O bei jedem Zyklus solange aktiv hält, bis %I00011 durchgeschaltet wird. Würde dieser Eingang nicht mit DO I/O abgefragt, nachdem SUS I/O aktiviert wurde, dann könnte SUS I/O nur durch Abschalten der SPS wieder deaktiviert werden.

Ausgang %Q00002 wird gesetzt, wenn beide DO I/O-Funktionen erfolgreich ausgeführt wurden. Der Strompfad ist so aufgebaut, daß beide DO I/O-Funktionen ausgeführt werden, selbst wenn bei einer der OK-Ausgang nicht gesetzt ist. Wenn die normale E/A-Aktualisierung unterdrückt ist, dann wird der Ausgang %Q00002 erst dann aktualisiert, wenn eine DO I/O-Funktion, die %Q00002 enthält, ausgeführt wird. Dies ist erst in dem Zyklus der Fall, der nach dem Setzen von %Q00002 abläuft. Ausgänge, die gesetzt werden, nachdem eine DO I/O-Funktion ausgeführt wurde, werden erst wieder aktualisiert, nachdem eine andere DO I/O-Funktion ausgeführt wurde, normalerweise im nächsten Zyklus. Durch diese Verzögerung wird bei den meisten Programmen, die SUS I/O und DO I/O verwenden, die SUS I/O-Funktion in den ersten Strompfad des Programms eingetragen, die DO I/O-Funktion, die die Eingänge bearbeitet, kommt in den nächsten Strompfad und die DO I/O-Funktion, die die Ausgänge bearbeitet, in den letzten Strompfad.

Der Bereich der DO I/O-Funktion, der die Ausgänge bearbeitet, liegt zwischen %Q00001 und %Q00030. Liegt in diesem Bereich z.B. ein 32-Punkt-Modul, dann führt die DO I/O-Funktion eine Aktualisierung des gesamten Moduls durch. Eine DO I/O-Funktion unterbricht nicht den Zyklus in der Mitte eines E/A-Moduls.



MCR

Sämtliche Strompfade zwischen einer aktiven Hauptsteuerrelaisfunktion MCR und der zugehörigen ENDMCR-Funktion werden mit negativer Logik ausgeführt. Mit der MCR-Funktion kann ein Teil der Programmlogik übergangen werden. Nach der zu MCR gehörenden Funktion ENDMCR wird die normale Programmausführung wieder aufgenommen. Im Gegensatz zu der JUMP-Funktion kann eine MCR-Funktion nur in Vorwärtsrichtung ausgeführt werden. Die Anweisung ENDMCR muß im Programm nach der entsprechenden MCR-Funktion stehen.

HINWEIS

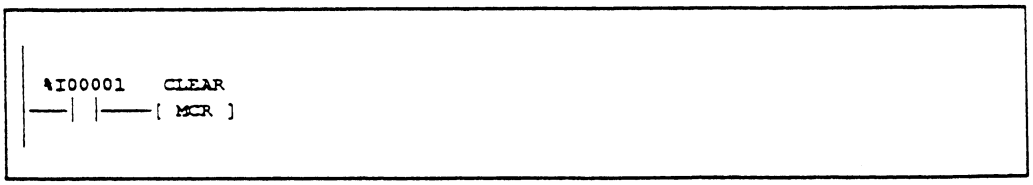
Programmblockaufrufe innerhalb des Wirkungsbereichs einer aktiven MCR-Funktion werden nicht ausgeführt. Zeitglieder im Programmblock werden jedoch weitergeführt.

Zu einer ENDMCR-Funktion ist nur jeweils eine MCR-Funktion möglich. Der Wirkungsbereich einer MCR-ENDMCR-Kombination darf den Wirkungsbereich einer anderen MCR-ENDMCR- oder JUMP/LABEL-Kombination nicht überschneiden. Eine MCR-Funktion darf sich nicht innerhalb des Wirkungsbereiches einer anderen MCR-ENDMCR- oder JUMP/LABEL-Kombination befinden. Darüberhinaus darf sich keine MCR-ENDMCR- oder JUMP/LABEL-Kombination innerhalb des Wirkungsbereiches eines MCR-ENDMCR-Paares befinden.

Die MCR-Funktion besitzt einen Boolschen Eingang (EN) sowie eine Bezeichnung, die die MCR-Funktion identifiziert. Diese Bezeichnung wird anschließend wieder mit der Funktion ENDMCR verwendet. Die MCR-Funktion besitzt keine Ausgänge, daher ist nach einer MCR-Funktion keine weitere Funktion im Strompfad mehr möglich.

Beispiel:

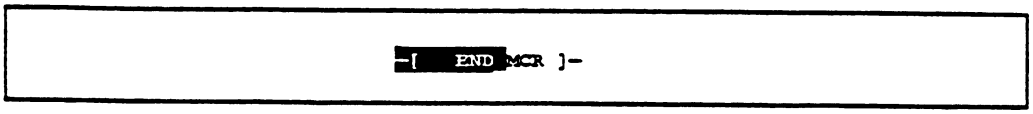
Wird bei dem folgenden Beispiel %I00001 durchgeschaltet, dann wird die Programmausführung bis zur zugeordneten Funktion END MCR mit negativem Stromfluß fortgesetzt.



ENDMCR

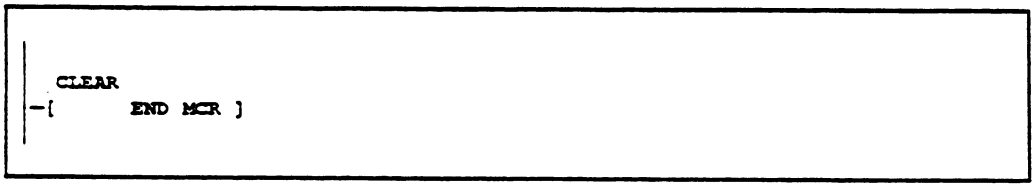
Mit ENDMCR wird nach einer MCR-Funktion wieder der normale Programmablauf aufgenommen. Nach einer aktiven MCR-Funktion veranlaßt die zugehörige ENDMCR-Funktion, daß das Programm wieder mit normalem Stromfluß fortgesetzt wird. War die zugehörige MCR-Funktion nicht aktiv, dann hat die ENDMCR-Funktion keine Wirkung.

Die ENDMCR-Funktion besitzt einen negierten Boolschen Eingang (EN). Die Freigabeanweisung muß von der Stromschiene kommen, eine bedingte Ausführung ist nicht zulässig. Die ENDMCR-Funktion wird durch einen Namen gekennzeichnet, der auch den Zusammenhang mit der entsprechenden MCR-Funktion herstellt. Die ENDMCR-Funktion besitzt keine Ausgänge, daher ist nach einer ENDMCR-Funktion keine weitere Funktion im Strompfad mehr möglich.



Beispiel:

Die ENDMCR-Funktion im folgenden Beispiel beendet den MCR-Bereich "CLEAR".

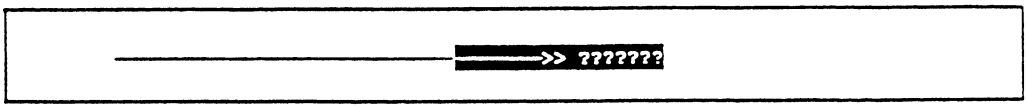


JUMP

Mit der JUMP-Funktion kann ein Teil des Programms in Vorwärts- oder Rückwärtsrichtung übersprungen werden. Die Programmausführung wird an der durch LABEL markierten Stelle fortgesetzt.

Die JUMP-Funktion hat Ähnlichkeit mit der MCR-Funktion, jedoch werden die Spulen innerhalb des Wirkungsbereiches von JUMP nicht mit negativer Logik bearbeitet. Ist JUMP aktiv, dann werden alle Spulen innerhalb des Wirkungsbereiches der Funktion eingefroren. Hierzu gehören auch Merker, die mit Zeitgliedern, Zählern, Haftrelais und Relais verbunden sind.

Die JUMP-Funktion wird immer in die Spalten 9 und 10 der aktuellen Strompfadzeile eingetragen. Nach einer JUMP-Funkktion sind keine weiteren Elemente mehr möglich. Der Stromfluß springt direkt von der Funktion zu dem mit LABEL gekennzeichneten Strompfad.

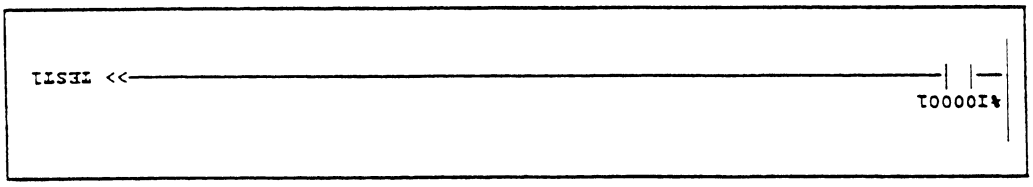


ACHTUNG

Um Endlosschleifen mit Vorwärts- und Rückwärstsprüngen zu vermeiden, muß ein Rückwärtssprung immer eine Bedingung enthalten.

Beispiel:

Ist im folgenden Beispiel JUMP TEST1 aktiv, dann wird der Stromfluß zu LABEL TEST1 weitergeschaltet.



LABEL

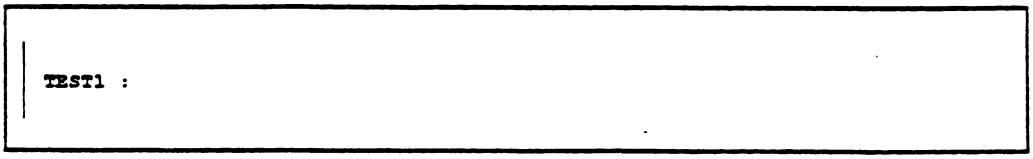
Die LABEL-Funktion bildet die Zielanweisung einer JUMP-Funktion. Mit ihr wird die normale Programmbearbeitung nach einer JUMP-Funktion wieder aufgenommen.

Ein Programm darf jeweils nur eine LABEL-Funktion mit einer bestimmten Bezeichnung enthalten. Programme, bei denen JUMP/LABEL-Kombinationen nicht geschlossen sind, können zwar erstellt und gespeichert werden, sind jedoch nicht ablauffähig.

Die LABEL-Anweisung, die nur für sich allein in einem Strompfad stehen darf, besitzt weder Eingangs- noch Ausgangsparameter.

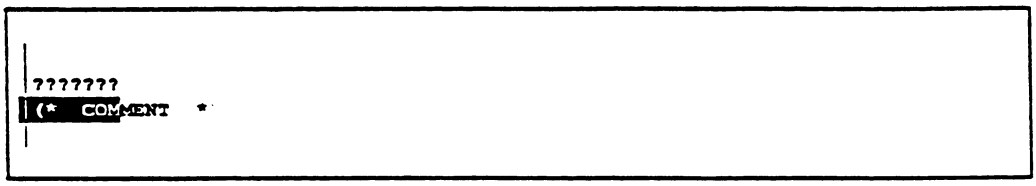
Beispiel:

Im folgenden Beispiel wird der Stromfluß von JUMP TEST1 bei LABEL TEST1 wieder aufgenommen.



COMMENT

Mit der COMMENT-Funktion können Erläuterungen (Strompfadkommentare) in ein Programm eingefügt werden. Ein Kommentar kann dabei bis zu 2048 Zeichen umfassen. Die entsprechende Darstellung im Kontaktplanprogramm ist:



Sie können den Text lesen oder editieren, wenn Sie den Cursor auf (* COMMENT *) setzen und nach der Übernahme des Strompfades die Taste F10 (Zoom) drücken. Kommentartexte können auch ausgedruckt werden. Bei längeren Kommentaren kann der Zusatztext auch in einer anderen Datei abgelegt werden. Weitere Informationen finden Sie in GFK-0263.

SVCREQ

Mit der SVCREQ-Funktion können Sie einen der folgenden SPS-Spezialdienste anfordern:

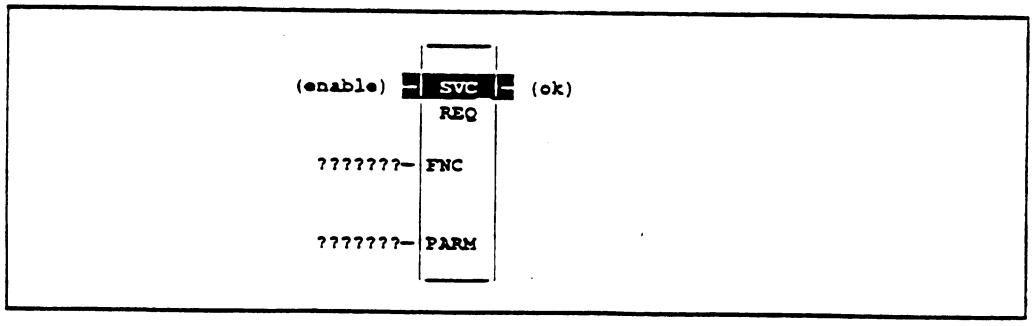
Tabelle 11-1. SVCREQ-Funktionen

Funktion	Beschreibung
1	Zeitglied für konstante Zyklusdauer lesen/stellen
2	Windowwerte lesen
3	Zustand und Werte im Programmiergeräte-Kommunikationswindow ändern
4	Zustand und Werte im System-Kommunikationswindow ändern
5	Reserviert
6	Reserviert
7	Echtzeituhr lesen/stellen
8	Zeitüberwachung (Watchdog) rücksetzen
9	Zykluszeit seit Zyklusbeginn lesen
10	Programmbezeichnung des aktuellen Blocks lesen
11	Steuerungskennung lesen
12	SPS-RUN-Status lesen
13	SPS abschalten
14	Fehlertabellen löschen
15	Letzten Fehlertabelleneintrag lesen
16	Betriebszeituhr lesen
17	E/A-Interrupt maskieren/entmaskieren
18	E/A-Override-Zustand lesen
19	RUN freigeben/sperren
20	Fehlertabellen lesen

Die SVCREQ-Funktion besitzt drei Eingangs- und einen Ausgangsparameter. Erhält die SVCREQ-Funktion Stromfluß, dann wird die SPS aufgefordert, die angegebenen Funktion (FNC) auszuführen. Die Parameter für die Funktion beginnen bei der für PARM angegebenen Referenz. Die SVCREQ-Funktion gibt Stromfluß weiter, sofern nicht eine falsche Funktionsnummer, falsche Parameter oder außerhalb des Bereichs liegende Referenzen angegeben wurden. Weitere Fehlerursachen werden auf den nächsten Seiten beschrieben.

Die für PARM angegebene Referenz muß in einem Wortspeicher (%R, %L, %P, %AI oder %AQ) liegen. Sie ist die erste einer Gruppe, die den Parameterblock der Funktion bilden. In aufeinanderfolgenden 16-Bit-Speicherplätzen werden weitere Parameter abgelegt. Die Gesamtzahl der benötigten Referenzen hängt von dem eingesetzten SVCREQ-Funktionstyp ab.

Parameterblöcke können sowohl als Eingänge für die Funktion als auch als die Stelle, an der die Daten nach Funktionsausführung abgelegt werden, verwendet werden. Auf die von der Funktion ausgegebenen Daten kann daher auch an der durch PARM angegebenen Stelle zugegriffen werden.



Parameter:

Parameter	Beschreibung
enable	Der gewünschte Bedienaufruf wird ausgeführt, wenn der ENABLE-Eingang aktiviert wird.
FNC	In FNC steht die Konstante oder Referenz für den gewünschten Dienst.
PARM	PARM enthält die Anfangsreferenz des Parameterblocks für den gewünschten Dienst.
ok	Der OK-Ausgang wird durchgeschaltet, wenn die Funktion normal ausgeführt wird.

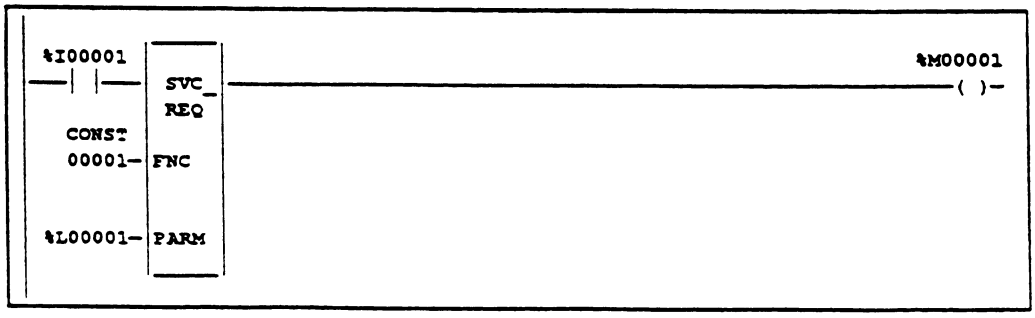
Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
FNC		•	•	•	•		•	•	•	•	•	•	•	
PARM		•	•	•	•		•	•	•	•	•	•		
ok	•													•

Hinweis: Bei allen Registerrefenzen ist indirekte Adressierung möglich (%R, %AI, %AQ, %P und %L).
• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

Beispiel:

Wird im folgenden Beispiel der Freigabeeingang %I00001 "wahr", dann wird die SVCREQ-Funktion Nummer 1 mit dem bei %L00001 beginnenden Parameterblock aufgerufen. Die Ausgangsspule %M00001 wird durchgeschaltet, wenn die Operation erfolgreich abläuft.



SVCREQ #1 Zeitglied für konstante Zyklusdauer lesen/stellen

Mit der SVCREQ-Funktion 1 können Sie:

- Die konstante Zyklusdauer deaktivieren.
- Konstante Zyklusdauer aktivieren und den alten Zeitwert verwenden.
- Konstante Zyklusdauer aktivieren und einen neuen Zeitwert verwenden.
- Nur einen neuen Zeitwert einstellen.
- Zustand und Zeitwert von konstanter Zyklusdauer lesen.

Bei den Funktionen für konstante Zyklusdauer besitzt der Parameterblock eine Länge von 2 Worten. Die Formate des Parameterblocks sind auf den folgenden Seiten dargestellt.

Die Funktion wird erfolgreich durchgeführt, sofern nicht:

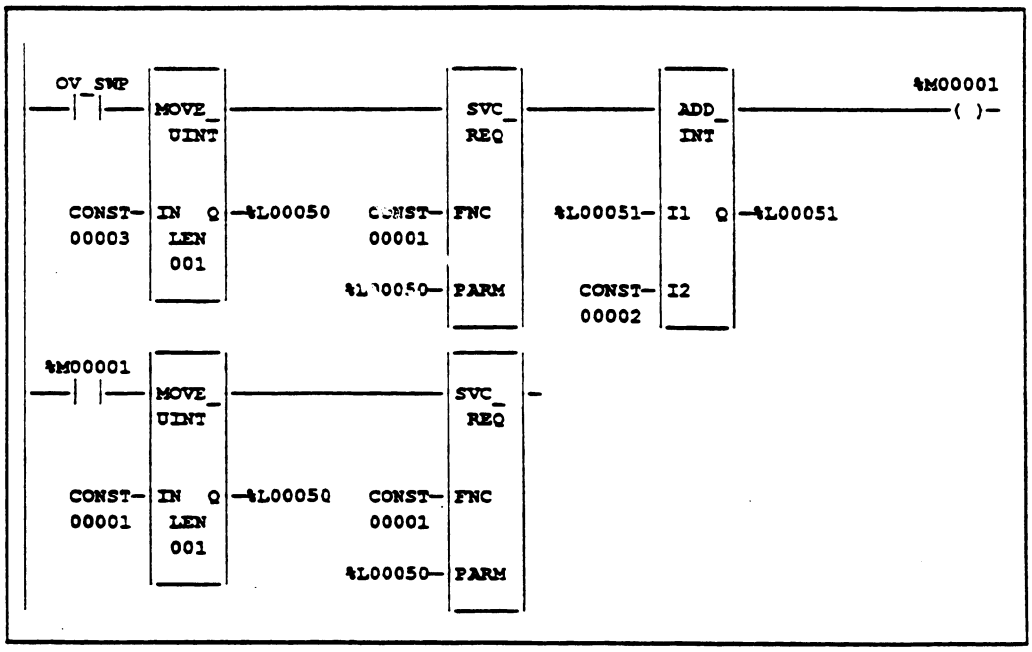
1. Eine andere Zahl als 0, 1, 2 oder 3 als gewünschte Operation eingegeben wird.

0	Konstante Zyklusdauer deaktivieren
1	Konstante Zyklusdauer aktivieren
2	Nur einen neuen Zeitwert einstellen
3	Zustand und Zeitwert von konstanter Zyklusdauer lesen

2. Der Zeitwert größer als 2550 ms (2,55 Sekunden) ist.
3. Die konstante Zyklusdauer wurde aktiviert, ohne daß ein neuer Zeitwert eingegeben wurde oder während ein alter Zeitwert 0 anstand.

Beispiel:

Dieses Beispiel zeigt Logik in einem Programmblock. Wird der Freigabekontakt OVSWP gesetzt, dann wird das Zeitglied für konstante Zyklusdauer gelesen, das Zeitglied um zwei Millisekunden erhöht und der neue Zeitwert zur SPS zurückgeschickt. Der Parameterblock steht im Lokalspeicher bei der Adresse %L00050. Da die Funktionen MOVE und ADD drei horizontale Kontaktpositionen belegen, wird in dem Programmbeispiel der diskrete interne Merker %M00001 als temporäre Adresse zur Aufnahme des erfolgreichen Ergebnisses aus der ersten Strompfadzeile verwendet. %M00001 wird in jedem Zyklus, in dem OV_SWP nicht gesetzt ist, abgeschaltet.



Konstante Zyklusdauer deaktivieren

Geben Sie eine SVCREQ-Funktion 1 mit dem folgenden Parameterblock ein:

0	Adresse
wird nicht beachtet	Adresse

Konstante Zyklusdauer aktivieren

Geben Sie eine SVCREQ-Funktion 1 mit dem folgenden Parameterblock ein:

1	Adresse
0 oder Zeitgliedwert	Adresse

Soll der Wert des Zeitglieds nicht verändert werden, dann geben Sie im zweiten Wort 0 ein. Existiert jedoch noch kein Zeitwert, dann wird durch die Eingabe von 0 der OK-Ausgang auf "falsch" gesetzt.

Geben Sie den neuen Wert im zweiten Wort ein, *wenn Sie den Wert des Zeitglieds verändern wollen.*

Nur einen neuen Zeitwert einstellen

Geben Sie eine SVCREQ-Funktion 1 mit dem folgenden Parameterblock ein, wenn Sie nur den Wert des Zeitgliedes ändern wollen, ohne den Betriebszustand zu verändern:

2	Adresse
neuer Zeitgliedwert	Adresse

Zustand und Zeitwert von konstanter Zyklusdauer lesen

Geben Sie eine SVCREQ-Funktion 1 mit dem folgenden Parameterblock ein, wenn sie nur den aktuellen Zustand und Zeitwert lesen wollen, ohne dabei Veränderungen vorzunehmen:

3	Adresse
wird nicht beachtet	Adresse

Nachdem die Funktion ausgeführt wurde, gibt sie den Zustand und Zeitwert unter der gleichen Parameterblockadresse aus:

0 = inaktiv 1 = aktiv	Adresse
aktueller Zeitwert	Adresse

Enthält das Wort n+1 den Hexadezimalwert FFFF, dann wurde noch kein Zeitwert programmiert.

SVCREQ #2 Windowwerte lesen

Mit der SVCREQ-Funktion 2 können Sie die aktuellen Windowzeiten anzeigen für das Programmiergeräte-Kommunikationswindow, das System-Kommunikationswindow und das Hintergrund-Window. Für jedes Window gibt es drei Betriebsarten:

Modus	Wert	Beschreibung
Begrenzt	0	Die Bearbeitungszeit des Windows ist beschränkt auf den jeweiligen Standardwert oder auf einen Wert, der mit der SVCREQ-Funktion 3 für das Programmiergeräte-Window, mit der SVCREQ-Funktion 4 für das Systemkommunikations-Window bzw. mit der SVCREQ-Funktion 5 für das Hintergrundaufgaben-Window eingestellt wird. Das Window wird abgeschlossen, wenn keine weiteren Aufgaben mehr zu erledigen sind.
Konstant	1	Jedes Window läuft in der Betriebsart "Läuft bis Abschluß" und die SPS wechselt über eine Zeitdauer, die der Summe der jeweiligen Window-Zeitwerte entspricht, zwischen den drei Windows. Die restlichen beiden Windows werden automatisch auf konstante Betriebsart eingestellt, wenn diese für ein Window eingestellt wurde. Arbeitet die SPS in konstanter Window-Betriebsart und wurde eine Window-Ausführungszeit nicht mit der entsprechenden SVCREQ-Funktion definiert, dann wird bei der Berechnung der konstanten Windowzeit für dieses Window die Standardzeit verwendet.
Läuft bis Abschluß	2	Unabhängig von der Zeit (Standardzeit oder über SVCREQ-Funktion zugewiesene Zeit), die einem bestimmten Window zugeteilt wurde, läuft das Window solange, bis alle Aufgaben innerhalb des betreffenden Windows abgeschlossen wurden.

Ein Window ist deaktiviert, wenn die Betriebsart begrenzt (0) und der Wert ebenfalls 0 ist.

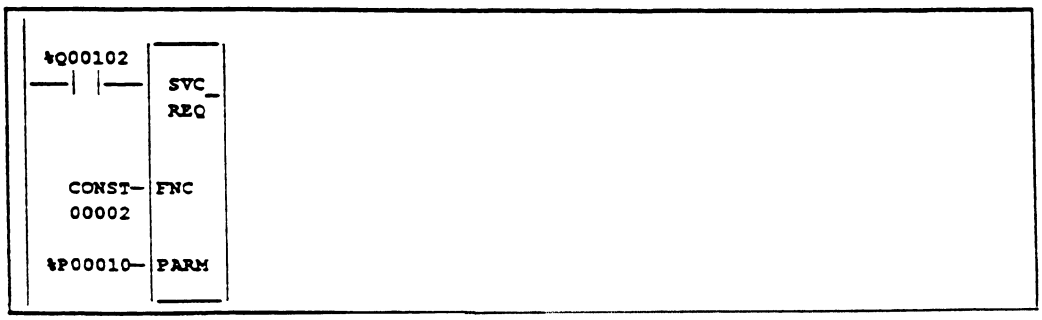
Bei den Funktionen zum Lesen des Windowwertes besitzt der Parameterblock eine Länge von 3 Worten.

	Oberes Byte	Unteres Byte	
Programmiergeräte-Window	Betriebsart	Wert in ms	Adresse
System-Kommunikationswindow	Betriebsart	Wert in ms	Adresse + 1
Hintergrund-Window	Betriebsart	Wert in ms	Adresse + 2

Sämtliche Parameter sind Ausgabeparameter. Bei der Programmierung dieser Funktion brauchen keine Werte in den Parameterblock eingetragen zu werden. Die Ausgangswerte für alle drei Windows werden in Millisekunden angegeben.

Beispiel:

Wird im folgenden Beispiel der Freigabeausgang %Q00102 durchgeschaltet, dann schreibt die SPS die aktuellen Zeitwerte der drei Windows in den Parameterblock, der bei Adresse %P00010 beginnt. Weitere Beispiele, die die Funktion "Windowwerte lesen" beinhalten, finden Sie in den nächsten drei Funktionsbeschreibungen.



SVCREQ #3 Zustand und Werte im Programmiergeräte-Kommunikationswindow ändern

Mit der SVCREQ-Funktion 3 können Sie das Programmiergeräte-Window aktivieren oder deaktivieren. Die Veränderung wirkt sich im gleichen CPU-Zyklus aus, in dem die Funktion aufgerufen wird.

Ist das Window freigegeben, dann gibt die Funktion den aktuellen Zeitwert für das Window zurück. Ist das Window deaktiviert, dann bleibt es so bis zur nächsten Freigabe. Die SVCREQ-Funktion #3 schaltet den Stromfluß immer nach rechts durch.

Der Parameterblock der SVCREQ-Funktion #3 besitzt eine Länge von einem Wort.

Geben Sie die SVCREQ-Funktion 3 und den folgenden Parameterblock ein, um das Programmiergeräte-Window zu deaktivieren:

Oberes Byte	Unteres Byte	
0	0	Adresse

Geben Sie die SVCREQ-Funktion 3 und den folgenden Parameterblock ein, um das Programmiergeräte-Window zu aktivieren:

Oberes Byte	Unteres Byte	
Modus	Wert zwischen 1 und 255 ms	Adresse

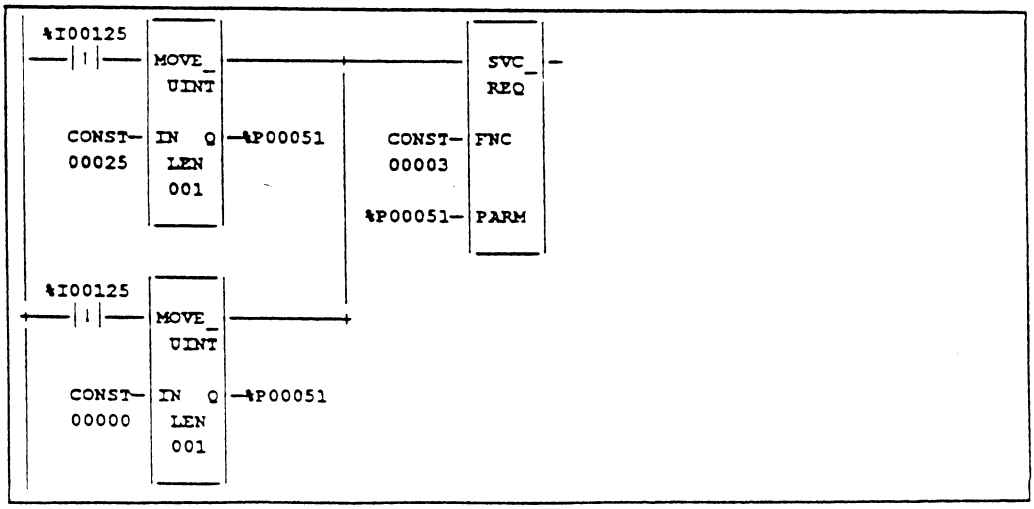
Aktivieren des Programmiergeräte-Windows:

Geben Sie die SVCREQ-Funktion 3 und den folgenden Parameterblock ein:

Oberes Byte	Unteres Byte	
Betriebsart	Wert zwischen 1 und 255 ms	Adresse

Beispiel:

Geht im folgenden Beispiel der Freigabeeingang %I00125 auf EIN, dann wird das Programmiergeräte-Window aktiviert und ihm ein Wert von 25 ms zugewiesen. Geht der Eingang auf AUS, dann wird das Window deaktiviert. Der Parameterblock steht in Adresse %P00051 im Globalspeicher.



SVCREQ #4 Zustand und Werte im System-Kommunikationswindow ändern

Mit der SVCREQ-Funktion 4 können Sie das System-Kommunikationswindow aktivieren und deaktivieren. Die Veränderung wirkt sich im gleichen CPU-Zyklus aus, in dem die Funktion aufgerufen wird. Ist das Window freigegeben, dann gibt die Funktion den aktuellen Zeitwert für das Window zurück. Ist das Window deaktiviert, dann bleibt es so bis zur nächsten Freigabe. Die SVCREQ-Funktion #4 schaltet den Stromfluß immer nach rechts durch.

Der Parameterblock der SVCREQ-Funktion #4 besitzt eine Länge von einem Wort.

Geben Sie die SVCREQ-Funktion 4 und den folgenden Parameterblock ein, wenn Sie das Systemkommunikations-Window deaktivieren wollen:

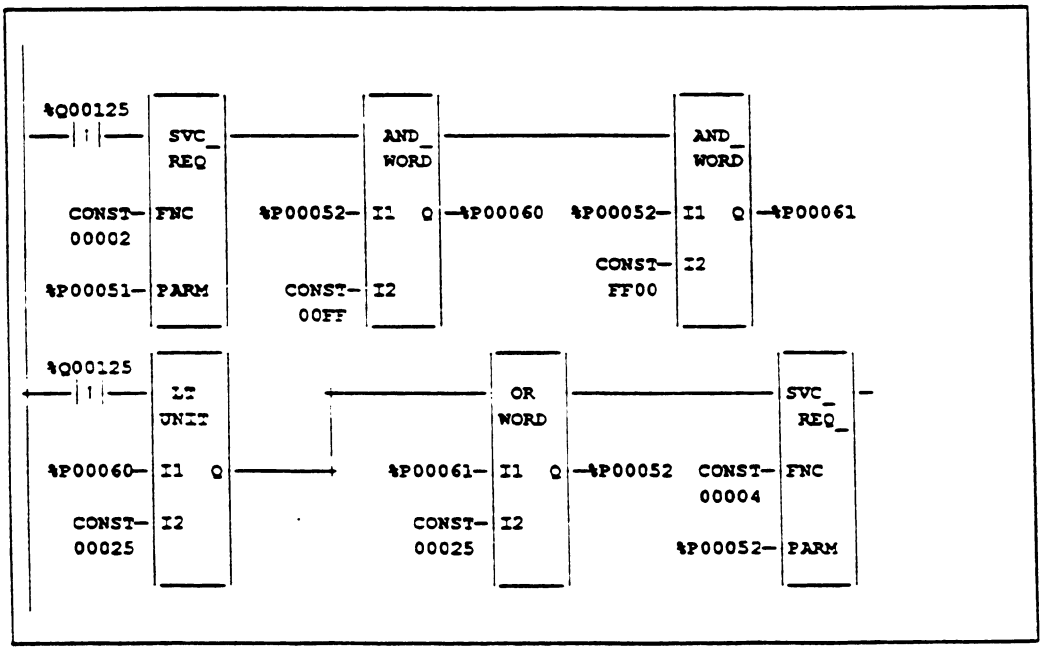
0	0	Adresse

Geben Sie die SVCREQ-Funktion 4 und den folgenden Parameterblock ein, wenn Sie das Systemkommunikations-Window aktivieren wollen:

Betriebsart	Wert zwischen 1 und 255 ms	
		Adresse

Beispiel:

Wird im folgenden Beispiel der Freigabeausgang %Q00125 durchgeschaltet, dann wird der Zeitwert des System-Kommunikationswindows gelesen. Ist dieser Wert nicht kleiner als 25 ms, dann bleibt er unverändert. Ist er jedoch kleiner als 25 ms, dann wird er auf 25 ms eingestellt. In beiden Fällen wird das Window bei der Strompfadbearbeitung aktiviert. Der Parameterblock für alle drei Windows steht bei Adresse %P00051. Da der Zeitwert für das System-Kommunikationswindow in dem von der SVCREQ Funktion #2 übertragenen Parameterblock an zweiter Stelle steht, steht die momentan eingestellte Zeitdauer für das System-Kommunikationswindow in %P00052.



SVCREQ #7 Echtzeituhr lesen/stellen

Mit der SVCREQ-Funktion 7 können Sie die Echtzeituhr in der SPS lesen oder einstellen.

Die Ausführung ist erfolgreich, mit Ausnahme von:

1. Eine von 0 oder 1 verschiedene Zahl wurde als Operationskennung eingegeben (siehe unten).
2. Es wurde ein unzulässiges Datenformat angegeben.
3. Die angebotenen Daten sind nicht im erwarteten Format.

Bei den Datum- und Zeitfunktionen hängt die Länge des Parameterblocks vom Datenformat ab. Numerische und unverdichtete BCD-Daten benötigen 9 Worte, BCD-Format benötigt 6 Worte, verdichtetes ASCII-Format 12 Worte.

0 = Zeit und Datum lesen 1 = Zeit und Datum einstellen	Adresse
0 = numerisches Datenformat 1 = BCD-Format 2 = unverdichtetes BCD-Format 3 = verdichtetes ASCII-Format	Adresse + 1
Daten	Adresse + 2 bis Ende

In Wort 1 wird festgelegt, ob die Funktion die Werte lesen oder verändern soll.

0 = lesen
1 = verändern

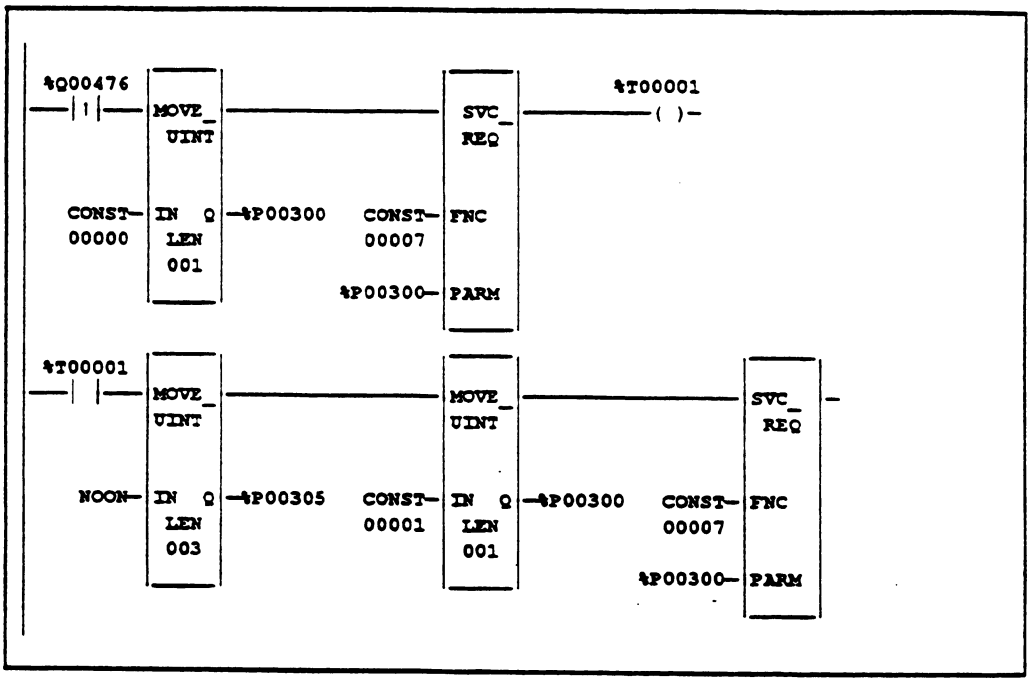
In Wort 2 wird das Datenformat angegeben:

0 = numerisch
1 = BCD-Format
2 = unverdichtetes BCD-Format
3 = verdichtetes ASCII-Format mit eingebetteten Leerzeichen und Doppelpunkten

Ab Wort 3 bis zum Ende des Parameterblocks werden die von der Lesefunktion ausgegebenen Daten oder neue, von der Veränderungsfunktion gelieferte Daten eingetragen. In beiden Fällen ist das Format dieser Datenworte gleich. Werden Datum und Uhrzeit gelesen, dann werden die Worte (n+2) bis (n+8) des Parameterblocks bei der Eingabe ignoriert.

Beispiel:

Wird beim folgenden Beispiel Ausgang %Q00476 durchgeschaltet, dann wird ein Parameterblock für die Echtzeituhr aufgebaut, der zunächst die aktuellen Werte von Datum und Uhrzeit ausliest und dann die Uhr im numerischen Format auf 12 Uhr Mittags einstellt. Der Parameterblock liegt an der Globaladresse %P00300. Das Feld NOON wurde an beliebiger Stelle im Programm eingerichtet und enthält die Werte 12, 0 und 0. Im BCD-Format werden neun aufeinanderfolgende Speicheradressen für den Parameterblock benötigt.



Inhalt des Parameterblocks

Auf den folgenden Seiten wird der Inhalt des Parameterblocks für die verschiedenen Datenformate dargestellt. Bei allen Datenformaten

- wird die Zeit im 24-Stunden-Format gespeichert;
- sind die Wochentage numerische Werte:

Wert	Wochentag
1	Sonntag
2	Montag
3	Dienstag
4	Mittwoch
5	Donnerstag
6	Freitag
7	Samstag

Datum und Zeit im numerischen Format ändern/lesen

Geben Sie die SVCREQ Funktion 7 mit nachstehendem Parameterblock ein. Im numerischen Format belegen Jahr, Monat, Tag, Stunden, Minuten, Sekunden und Wochentag jeweils eine ganze Zahl ohne Vorzeichen.

Unteres Byte	Oberes Byte		Beispiel Ausgangs-Parameterblock Datum und Zeit im BCD-Format lesen (Mittwoch, 3. Juni 1988, 12:15:30 Uhr)
0 = lesen	1 = ändern	Adresse	0
	0	Adresse + 1	0
Jahr		Adresse + 2	88
Monat		Adresse + 3	06
Tag		Adresse + 4	15
Stunden		Adresse + 5	12
Minuten		Adresse + 6	15
Sekunden		Adresse + 7	30
Wochentag		Adresse + 8	04

Daten im BCD-Format ändern/lesen

Im BCD-Format belegt jedes Zeit- und Datumselement ein einzelnes Byte. Dieses Format benötigt sechs Worte. Das letzte Byte des sechsten Wortes wird dabei nicht verwendet. Werden Datum und Zeit eingestellt, dann wird dieses Byte ignoriert; werden Datum und Zeit gelesen, dann gibt diese Funktion ein Nullzeichen zurück (00).

Unteres Byte	Oberes Byte		Beispiel Ausgangs-Parameterblock Datum und Zeit im BCD-Format lesen (Montag, 3. Juli 1988, 2:45:30 Uhr)	
0 = lesen	**1 = ändern**	**Adresse**	**0**	
	1	**Adresse + 1**	**1**	
Jahr	**Monat**	**Adresse + 2**	**89**	**07**
Tag	**Stunden**	**Adresse + 3**	**03**	**14**
Minuten	**Sekunden**	**Adresse + 4**	**45**	**30**
Wochentag	**(Null)**	**Adresse + 5**	**02**	**00**

Daten im unverdichteten BCD-Format ändern/lesen

In unverdichteten BCD-Format belegt jede Stelle von Zeit und Datum die unteren vier Bits eines Bytes. Die oberen vier Bits dieser Bytes sind immer Null. Dieses Format belegt neun Worte.

Unteres Byte	Oberes Byte		Beispiel Ausgangs-Parameterblock Datum und Zeit im BCD-Format lesen (Donnerstag, 28. Dezember 1989, 9:34:57 Uhr)	
0 = lesen	1 = ändern	Adresse	0	
	0	Adresse + 1	2	
Jahr		Adresse + 2	08	09
Monat		Adresse + 3	01	02
Tag		Adresse + 4	02	08
Stunden		Adresse + 5	00	09
Minuten		Adresse + 6	03	04
Sekunden		Adresse + 7	05	07
Wochentag		Adresse + 8	00	05

Daten im verdichteten ASCII-Format mit eingebetteten Doppelpunkten ändern/lesen

Im verdichteten ASCII-Format ist jede Stelle der Zeit- und Datumswerte ein Byte im ASCII-Format. Darüberhinaus sind Leerzeichen und Doppelpunkte in den Daten eingebettet, um eine unveränderte Übertragung zu einem Drucker oder Sichtgerät zu ermöglichen. Dieses Format belegt 12 Worte.

Unteres Byte	Oberes Byte		Beispiel Ausgangs-Parameterblock Datum und Zeit im BCD-Format lesen (Dienstag, 2. Okt. 1989, 23:13:00 Uhr)	
0 = lesen	1 = ändern	Adresse	0	
	3	Adresse + 1	3	
Jahr	Jahr	Adresse + 2	39/30	
(Leerzeichen)	Monat	Adresse + 3	20	31
Monat	(Leerzeichen)	Adresse + 4	30	20
Tag	Tag	Adresse + 5	30	32
(Leerzeichen)	Stunden	Adresse + 6	20	32
Stunden	:	Adresse + 7	33	3A
Minuten	Minuten	Adresse + 8	31	33
:	Sekunden	Adresse + 9	3A	30
Sekunden	(Leerzeichen)	Adresse + 10)	30	20
Wochentag	Wochentag	Adresse +11	30	33

SVCREQ #8 Zeitüberwachung (Watchdog) rücksetzen

Mit der SVCREQ-Funktion 8 können Sie während des Zyklus die Zeitüberwachung (Watchdog) lesen oder einstellen.

Läuft die Zeitüberwachung (Watchdog) ab, dann wird die SPS ohne Vorwarnung abgeschaltet. Mit der Funktion "Zeitüberwachung (Watchdog) rücksetzen" kann das Zeitglied über eine zeitaufwendige Aufgabe (zum Beispiel während der Wartezeit auf eine Antwort von einer Kommunikationsleitung) hinweg weitergeführt werden.

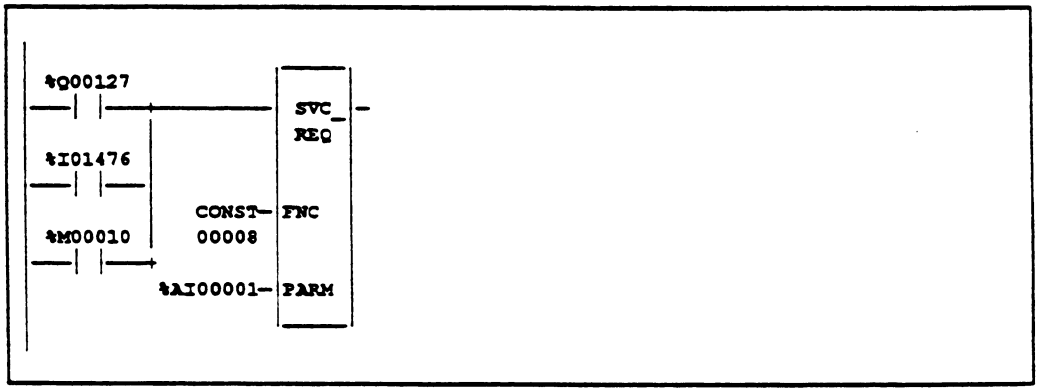
ACHTUNG

Stellen Sie sicher, daß das Rücksetzen der Zeitüberwachung keine nachteiligen Folgen für den gesteuerten Prozeß hat.

Obwohl die Funktion "Zeitüberwachung (Watchdog) rücksetzen" keinen Parameterblock besitzt, verlangt die Programmiersoftware, daß eine Eintragung für PARM vorgenommen wird. Geben Sie hier eine passende Referenz ein, die jedoch nicht verwendet wird.

Beispiel:

Werden im folgenden Beispiel Freigabeausgang %Q00127 oder Eingang %I01476 oder interner Merker %M00010 gesetzt, dann wird die Zeitüberwachung (Watchdog) rückgesetzt.



SVCREQ #9 Zykluszeit seit Zyklusbeginn lesen

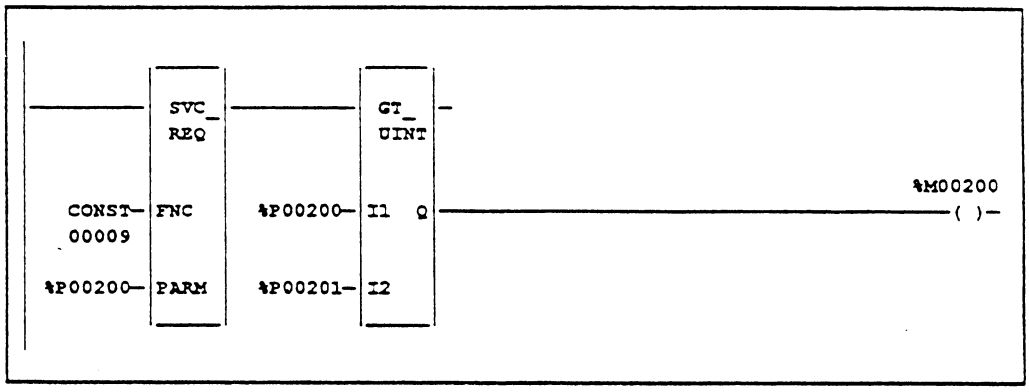
Mit der SVCREQ-Funktion 9 können Sie sich die seit Zyklusbeginn verstrichene Zeit in Millisekunden ausgeben lassen. Als Datenformat werden vorzeichenlose ganze Zahlen (16 Bit) verwendet.

Der Parameterblock der SVCREQ-Funktion #9 besitzt eine Länge von einem Wort.

Zeit seit Zyklusbeginn	Adresse

Beispiel:

Im folgenden Beispiel wird die seit Zyklusbeginn verstrichene Zeit immer in Adresse %P00200 eingelesen. Der interne Merker %M00200 wird durchgeschaltet, wenn dieser Wert größer ist als der in %P00201 gespeicherte Wert.



SVCREQ #10 Programmidentifikation des aktuellen Blocks lesen

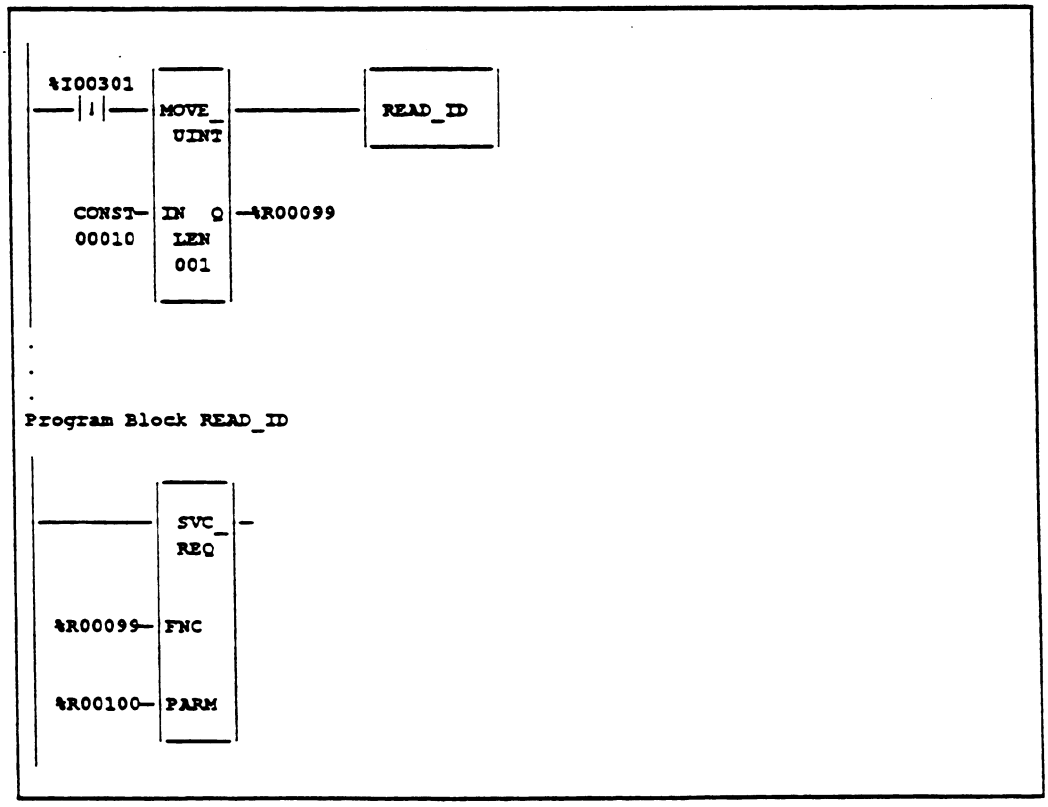
Mit der SVCREQ-Funktion 10 können Sie den Namen des momentan ablaufenden Programmblocks lesen.

Der Parameterblock der SVCREQ-Funktion #10, der eine Länge von vier Worten besitzt, gibt acht ASCII-Zeichen zurück. Das letzte dieser Zeichen ist Null. Besitzt die Programmbezeichnung weniger als 7 Zeichen, dann werden am Ende Nullen eingefügt.

Oberes Byte	Unteres Byte	
Zeichen 1	Zeichen 2	Adresse
Zeichen 3	Zeichen 4	Adresse + 1
Zeichen 5	Zeichen 6	Adresse + 2
Zeichen 7	00	Adresse + 3

Beispiel:

Geht im folgenden Beispiel der Freigabeeingang %I00301 auf AUS, dann wird der Wert 10 (der Funktionscode für die Funktion "Programmidentifikation lesen") in die Registeradresse %R00099 geladen. Der Programmblock READ_ID wird dann aufgerufen, der die Kennung ermitteln soll. Der Parameterblock liegt in Adresse %R00100. READ_ID wird auch im nächsten Beispiel verwendet.



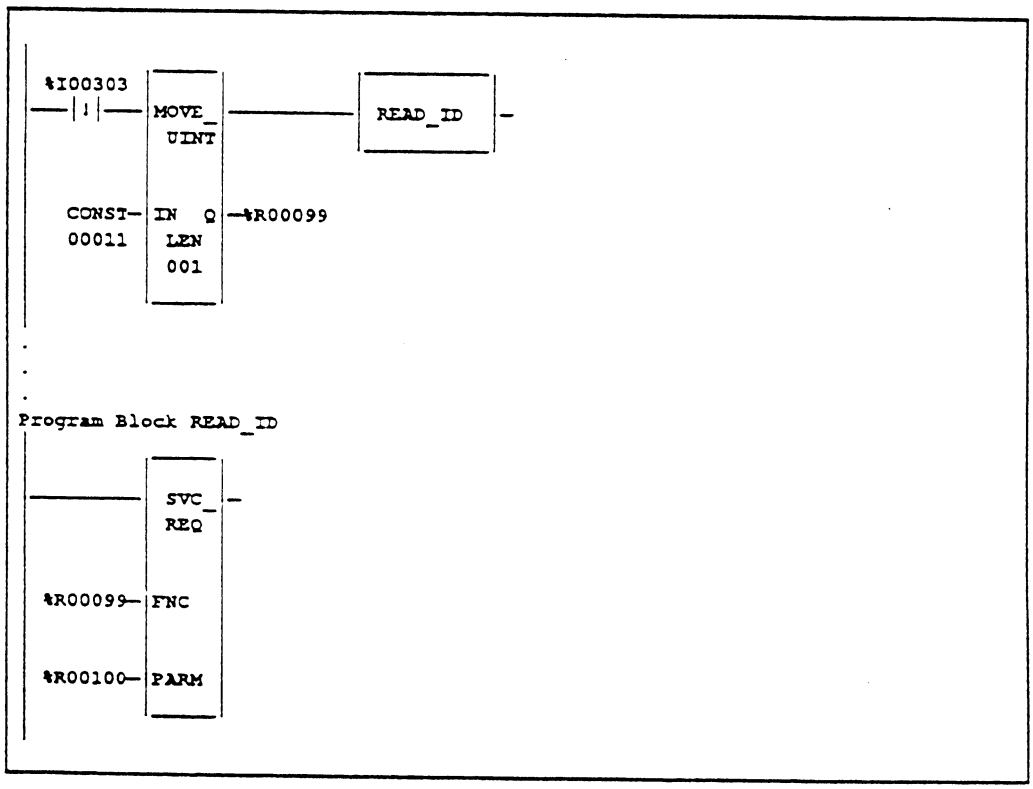
SVCREQ #11 Steuerungskennung lesen

Mit der SVCREQ-Funktion 6 können Sie die Kennung der SPS Serie 90 ermitteln, in der das Programm gerade läuft. Der Parameterblock der SVCREQ-Funktion #11, der eine Länge von vier Worten besitzt, gibt acht ASCII-Zeichen zurück. Das letzte dieser Zeichen ist Null. Besitzt die SPS-Kennung weniger als 7 Zeichen, dann werden am Ende Nullen eingefügt.

Oberes Byte	Unteres Byte	
Zeichen 1	Zeichen 2	Adresse
Zeichen 3	Zeichen 4	Adresse + 1
Zeichen 5	Zeichen 6	Adresse + 2
Zeichen 7	00	Adresse + 3

Beispiel:

Geht im folgenden Beispiel der Freigabeeingang %I00302 auf AUS, dann wird der Wert 11 (der Funktionscode für die Funktion "SPS-Kennung lesen") in die Registeradresse %R00099 geladen. Der Programmblock READ_ID wird dann aufgerufen, der die Kennung ermitteln soll. Der Parameterblock liegt in der Adresse %R00100. Mit Ausnahme von Freigabekontakt- und Funktionsnummer wird hier der gleiche Code verwendet wie im letzten Beispiel.



SVCREQ #12 SPS-RUN-Status lesen

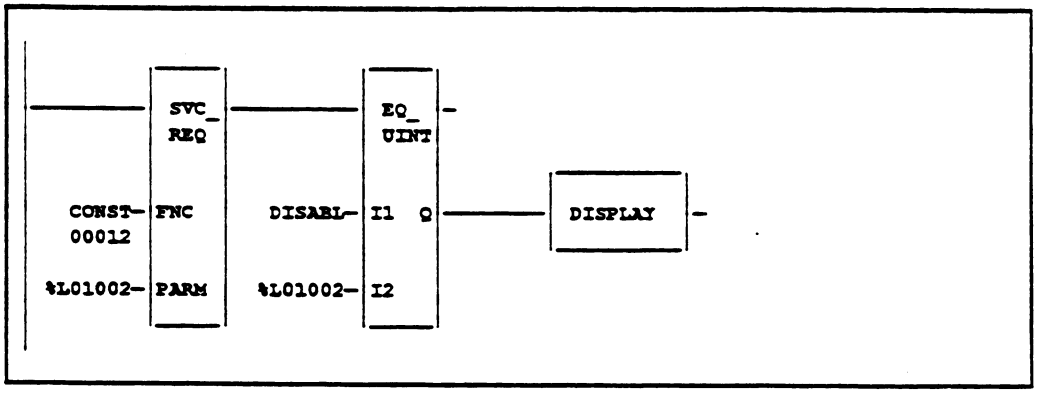
Mit der SVCREQ-Funktion 12 können Sie den aktuellen RUN-Status der SPS CPU lesen.

Der Parameterblock dieser SVCREQ-Funktion, der eine Länge von einem Worten besitzt, ist ein Ausgabeparameterblock.

1 = RUN/gesperrt 2 = RUN/freigegeben	Adresse

Beispiel:

Im folgenden Beispiel wird der RUN-Status der SPS immer in die Adresse %L01002 eingelesen. Ist der Status RUN/gesperrt, dann ruft die CALL-Funktion den Programmblock DISPLAY auf.



SVCREQ #13 SPS abschalten

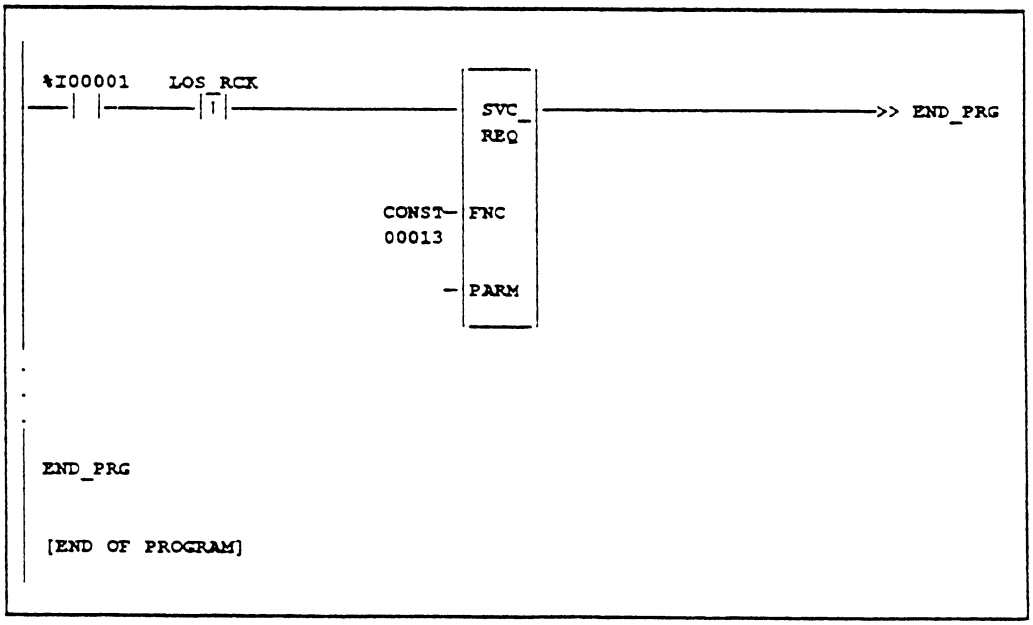
Mit der SVCREQ-Funktion 13 können Sie die SPS am Ende des aktuellen Zyklus in STOP-Modus versetzen. Die Ausgänge werden dann zu Beginn des nächsten SPS-Zyklus auf ihre vorgegebenen Zustände eingestellt. In der SPS-Fehlertabelle wird ein informativer Fehler eingetragen, der besagt, daß ein Funktionsblock "SPS abschalten" ausgeführt wurde.

Die Funktion besitzt keinen Parameterblock.

Beispiel:

Wird im folgenden Beispiel der Freigabeeingang %I00001 gesetzt, dann wird die SVCREQ-Funktion 13 ausgeführt, wenn ein Chassisverlust auftritt. Da kein Parameterblock benötigt wird, wird der PARM-Eingang nicht verwendet. Die Programmiersoftware verlangt jedoch, daß eine Eintragung für PARM vorgenommen wird.

Dieses Beispiel verwendet eine JUMP-Funktion zum Programmende, um eine Abschaltung zu erzwingen, wenn die Abschaltfunktion erfolgreich ausgeführt wurde. Diese JUMP- und LABEL-Anweisungen werden benötigt, da der Übergang in den STOP-Modus erst am Ende des Zyklus erfolgt, in dem die Funktion ausgeführt wird.



SVCREQ #14 Fehlertabellen löschen

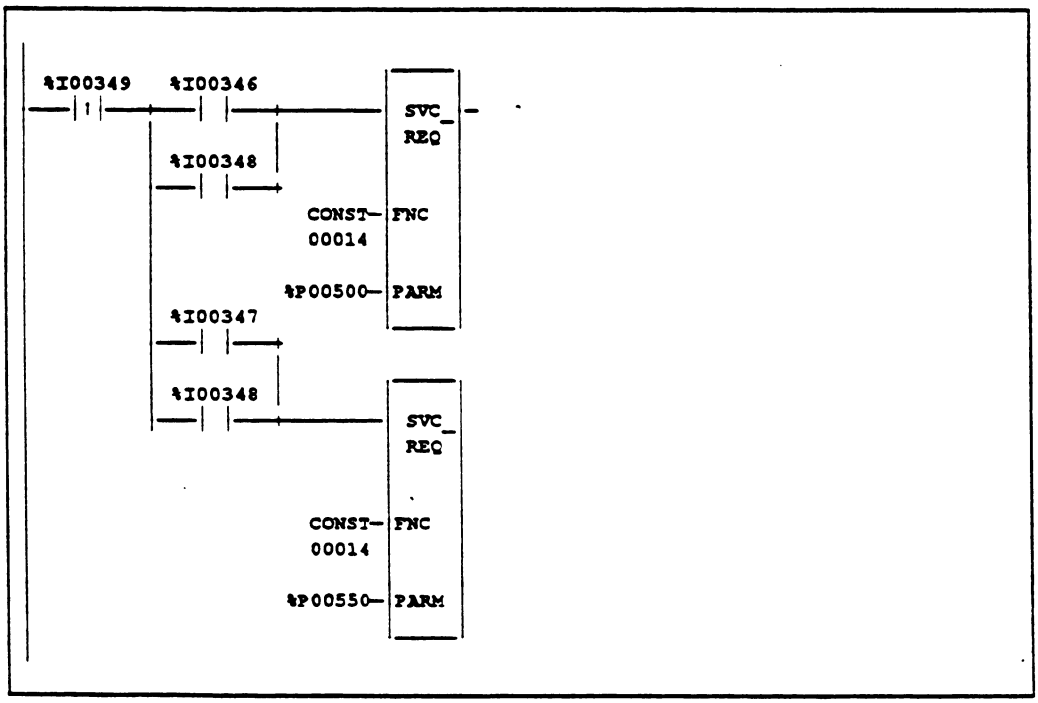
Mit der SVCREQ-Funktion 14 können SPS- oder E/A-Fehlertabelle gelöscht werden. Der Ausgang dieser SVCREQ-Funktion wird "wahr", sofern keine andere Zahl als 0 oder 1 für die gewünschte Operation eingegeben wird (siehe unten). Bei leerer Fehlertabelle schaltet die SVCREQ-Funktion den Stromfluß nicht durch.

Bei dieser Funktion ist der Parameterblock (nur Eingabe-Parameterblock) ein Wort lang.

0 = SPS-Fehlertabelle löschen 1 = E/A-Fehlertabelle löschen	Adresse

Beispiel:

Im folgenden Beispiel wird die SPS-Fehlertabelle gelöscht, wenn die Eingänge %I00346 und %I00349 durchgeschaltet sind. Die E/A-Fehlertabelle wird gelöscht, wenn die Eingänge %I00347 und %I00349 durchgeschaltet sind. Beide Tabellen werden gelöscht, wenn die Eingänge %I00348 und %I00349 durchgeschaltet sind. Der Parameterblock für die SPS-Tabelle liegt bei %R00500, für die E/A-Fehlertabelle bei %R00550. Beide Parameterblöcke liegen an einer anderen Stelle im Programm.



SVCREQ #15 Letzten Fehlertabelleneintrag lesen

Mit der SVCREQ-Funktion 15 können Sie den letzten Eintrag in die SPS- oder E/A-Fehlertabelle lesen. Der Ausgang dieser SVCREQ-Funktion wird "falsch", sofern keine andere Zahl als 0 oder 1 für die gewünschte Operation eingegeben wird (siehe unten). Bei leerer Fehlertabelle schaltet die SVCREQ-Funktion den Stromfluß nicht durch. (Weitere Informationen über die Einträge in den Fehlertabellen finden Sie in GFK-0262).

Bei dieser Funktion ist der Parameterblock 22 Worte lang. Der Eingabe-Parameterblock hat das Format:

0 = SPS-Fehlertabelle lesen 1 = E/A-Fehlertabelle lesen	Adresse

Das Format des Ausgabe-Parameterblocks hängt davon ab, ob die Funktion Daten aus der SPS- oder der E/A-Fehlertabelle liest.

Ausgabeformat SPS-Fehlertabelle

Unteres Byte	Oberes Byte		
0		Adresse	
lang/kurz		Adresse + 1	
Reserve		Adresse + 2	
SPS-Fehleradresse		Adresse + 3 Adresse + 4	
Fehlergruppe und Aktion		Adresse + 5	
Fehlercode		Adresse + 6	
Fehlerspezifische Daten		Adresse + 7 Adresse + 8 Adresse + 9 Adresse + 10 Adresse + 11 Adresse + 12 Adresse + 13 Adresse + 14 Adresse + 15 Adresse + 16 Adresse + 17 Adresse + 18	
Zeitmarkierung		Adresse + 19 Adresse + 20 Adresse + 21	

Ausgabeformat E/A-Fehlertabelle

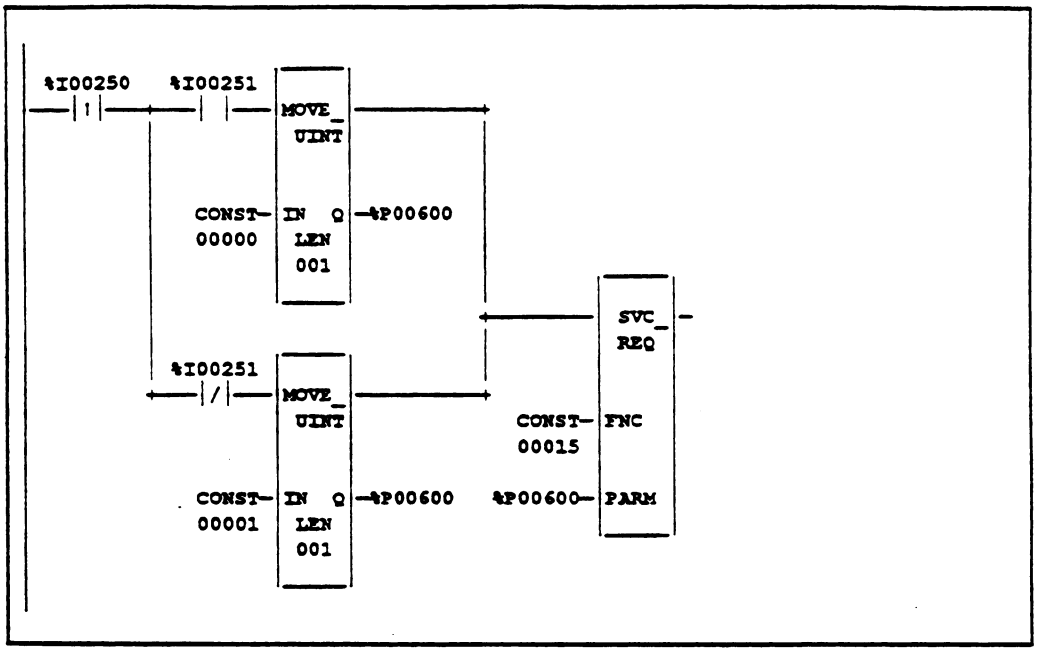
Unteres Byte	Oberes Byte
1	
lang/kurz	
Referenzadresse	
E/A-Fehleradresse	
Fehlergruppe und Aktion	
Fehlerkategorie	Fehlertyp
Fehlerbeschreibung	
Fehlerspezifische Daten	
Zeitmarkierung	

Im ersten Byte von Wort n+1 gibt "kurz/lang" die Anzahl der fehlerspezifischen Daten im Fehlereintrag an. Es bedeuten:

SPS-Fehlertabelle:	00 = 8 Bytes (kurz) 01 = 24 Bytes (lang)
E/A-Fehlertabelle:	02 = 5 Bytes (kurz) 03 = 21 Bytes (lang)

Beispiel 1:

Sind im folgenden Beispiel die Eingänge %I00251 und %I00250 durchgeschaltet, dann wird der letzte Eintrag der SPS-Fehlertabelle in den Parameterblock eingelesen. Wenn Eingang %I00251 AUS und Eingang %I00250 EIN sind, dann wird der letzte Eintrag der E/A-Fehlertabelle in den Parameterblock eingelesen. Der Parameterblock liegt im Globalspeicher bei der Adresse %=00600.



Beispiel 2:

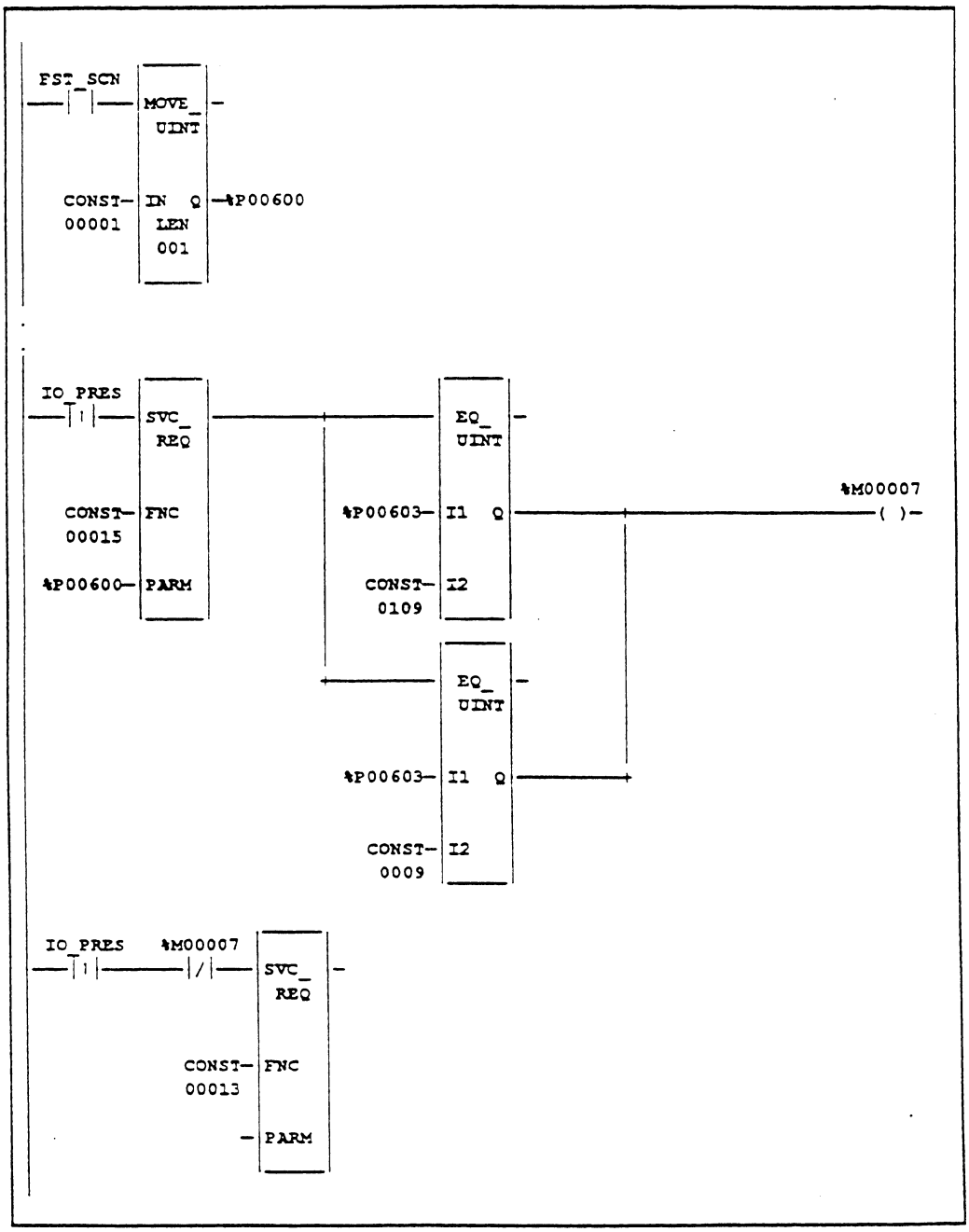
Im folgenden Beispiel wird die SPS abgeschaltet, wenn ein Fehler auf einem E/A-Modul auftritt (Ausnahme: Fehler auf Modulen in Chassis 0, Steckplatz 9 oder Chassis 1, Steckplatz 9). Tritt ein Fehler in einem dieser beiden Module auf, dann läuft das System weiter. Der Parameter für "Tabellentyp" wird im ersten Zyklus eingestellt. Ist der Kontakt IO_PRES gesetzt, dann enthält die E/A-Fehlertabelle einen Eintrag. Die SPS CPU setzt den Schließerkontakt in dem auf den Fehlereintrag folgenden Zyklus. Werden bei zwei aufeinanderfolgenden Zyklen Fehler eingetragen, dann wird der Kontakt für positiven Übergang bei zwei aufeinanderfolgenden Zyklen gesetzt.

Das Beispiel verwendet einen Parameterblock im Globalspeicher %P00600. Nachdem die SVCREQ-Funktion ausgeführt wurde, steht die Adresse des fehlerhaften Moduls im 4., 5. und 6. Wort des Parameterblocks:

1		%P00600
lang/kurz		%P00601
Referenzadresse		%P00602
Chassisnummer	Steckplatznummer	%P00603
E/A-Busnummer	Busadresse	%P00604
Punktadresse		%P00605

Fehlerdaten

Im Programm vergleichen die EQ_UINT-Blöcke die Chassis-/Steckplatzadresse in der Tabelle mit hexadezimalen Konstanten. Der interne Merker %M00007 wird durchgeschaltet, wenn Chassis/Steckplatz mit dem fehlerhaften Modul die vorgenannten Kriterien erfüllen. Ist %M00007 durchgeschaltet, dann ist sein Öffnerkontakt AUS und verhindert die Abschaltung. Ist dagegen %M00007 AUS, da der Fehler auf einem anderen Modul aufgetreten ist, dann ist der Öffnerkontakt EIN und die SPS wird abgeschaltet.



SVCREQ # 16 .i.Betriebszeituhr lesen

Mit der SVCREQ-Funktion 16 kann der Wert der Betriebszeituhr ausgelesen werden. Beginnend mit dem Zeitpunkt, zu dem die SPS eingeschaltet wird, erfaßt diese Uhr die Zeit in Sekunden. Der Erfassungszeitraum dieser Uhr beträgt ungefähr 100 Jahre.

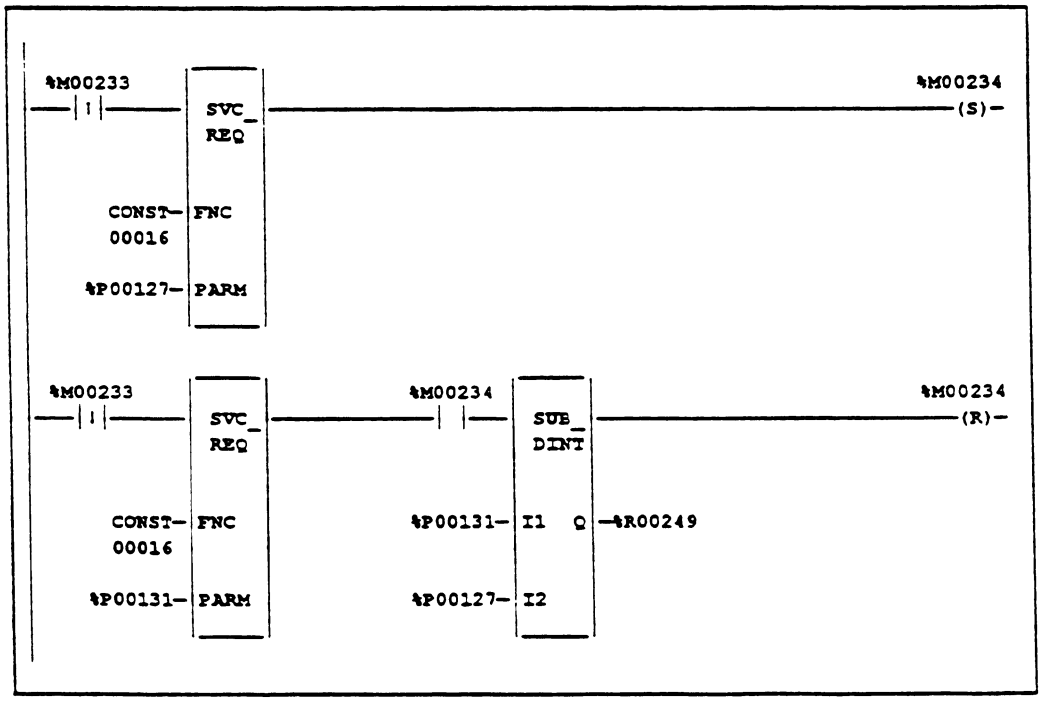
Die Funktion besitzt lediglich einen Ausgabe-Parameterblock mit einer Länge von 3 Worten.

Sekunden seit Einschalten (niedrigwertig)	Adresse
Sekunden seit Einschalten (hochwertig)	Adresse + 1
100-Mikrosekunden-Werte	Adresse + 2

Die beiden ersten Werte geben die Betriebszeit in Sekunden an. Das letzte Wort gibt den 100-Mikrosekunden-Teil der laufenden Sekunde an.

Beispiel:

Wird beim folgenden Beispiel der interne Merker %M00233 durchgeschaltet, dann wird der Wert der Betriebszeituhr gelesen und die Spule %M00234 gesetzt. Der Wert wird erneut gelesen, wenn der interne Merker abfällt. Der Unterschied zwischen den beiden Werten wird dann berechnet und das Ergebnis im Registerspeicher unter der Adresse %R00250 abgelegt. Der Parameterblock beim ersten Lesen ist %P00127, beim zweiten Lesen %P00131. Bei der Berechnung werden die 100-Mikrosekunden-Anteile und die Tatsache, daß der DINT-Type eigentlich ein vorzeichenbehafteter Wert ist, vernachlässigt. Die Berechnung ist über eine Einschaltdauer von etwa 50 Jahren korrekt.



SVCREQ #17 E/A-Interrupt maskieren/entmaskieren

Mit der SVCREQ-Funktion 17 kann ein Interrupt von einem Eingangsmodul maskiert oder entmaskiert werden. Wird ein Interrupt maskiert, dann wird der entsprechende Interruptblock von der CPU nicht bearbeitet, wenn der Eingang umschaltet und einen Interrupt erzeugt.

Die Ausführung ist erfolgreich, mit Ausnahme von:

1. Eine von 0 oder 1 verschiedene Zahl wurde als Operationskennung eingegeben (siehe unten).
2. Der Speichertyp des von der Maskierung betroffenen Eingangs ist nicht %I.
3. Das E/A-Modul ist kein passendes Eingangsmodul.
4. Auf dem Modul wurden die Interrupts deaktiviert.
5. Die angegebene Adresse ist kein maskierbarer Kanal eines Eingangsmodul der Serie 90.

Der Funktionsblock besitzt nur einen Eingangsparameterblock mit einer Länge von 3 Worten.

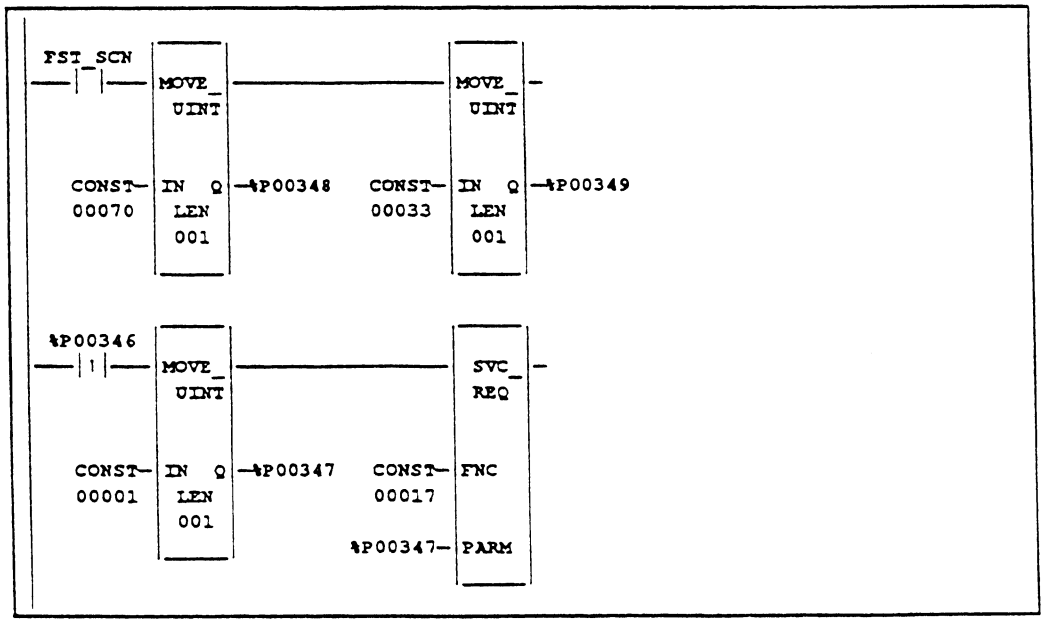
0 = Eingang entmaskieren 1 = Eingang maskieren	Adresse
Speichertyp	Adresse + 1
Adresse (Relativzeiger)	Adresse + 2

"Speichertyp" ist eine Dezimalzahl im unteren Byte des Wortes n+1, die dem Speichertyp des Eingangs entspricht:

70 = I-Speicher in Bitmodus.

Beispiel:

Wird im folgenden Beispiel %P00346 durchgeschaltet, dann werden Interrupts von Eingang %I00033 maskiert. Der Parameterblock bei %P00347 wird beim ersten Zyklus eingerichtet.



SVCREQ #18 E/A-Override-Zustand lesen

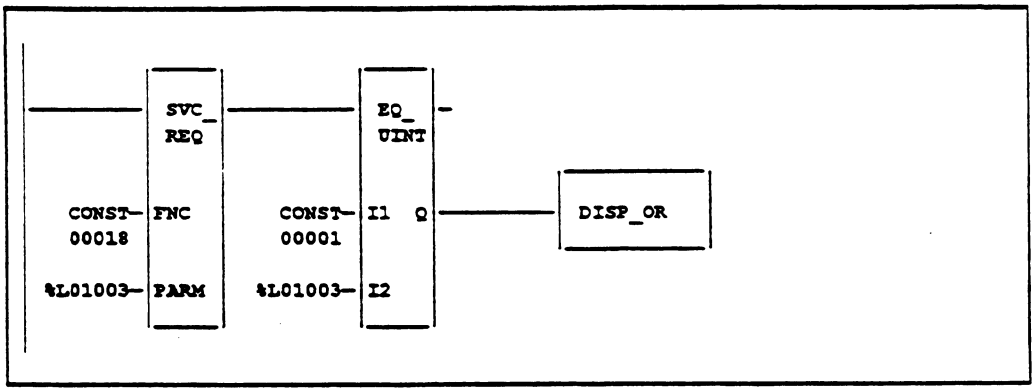
Mit der SVCREQ-Funktion 18 kann der aktuelle Override-Zustand in der CPU gelesen werden.

Bei dieser Funktion besitzt der Parameterblock (nur Ausgabe-Parameterblock) eine Länge von einem Wort.

0 = keine Overrides gesetzt 1 = Overrides gesetzt	Adresse

Beispiel:

Im folgenden Beispiel wird der Zustand der E/A-Overrides immer in Adresse %L01003 eingelesen. Der Programmblock DISP_OR wird aufgerufen, wenn Override-Zustände bestehen.



SVCREQ #19 RUN freigeben/sperren

Mit der SVCREQ-Funktion 19 kann mit dem Kontaktplanprogramm der RUN-Modus der CPU gesteuert werden.

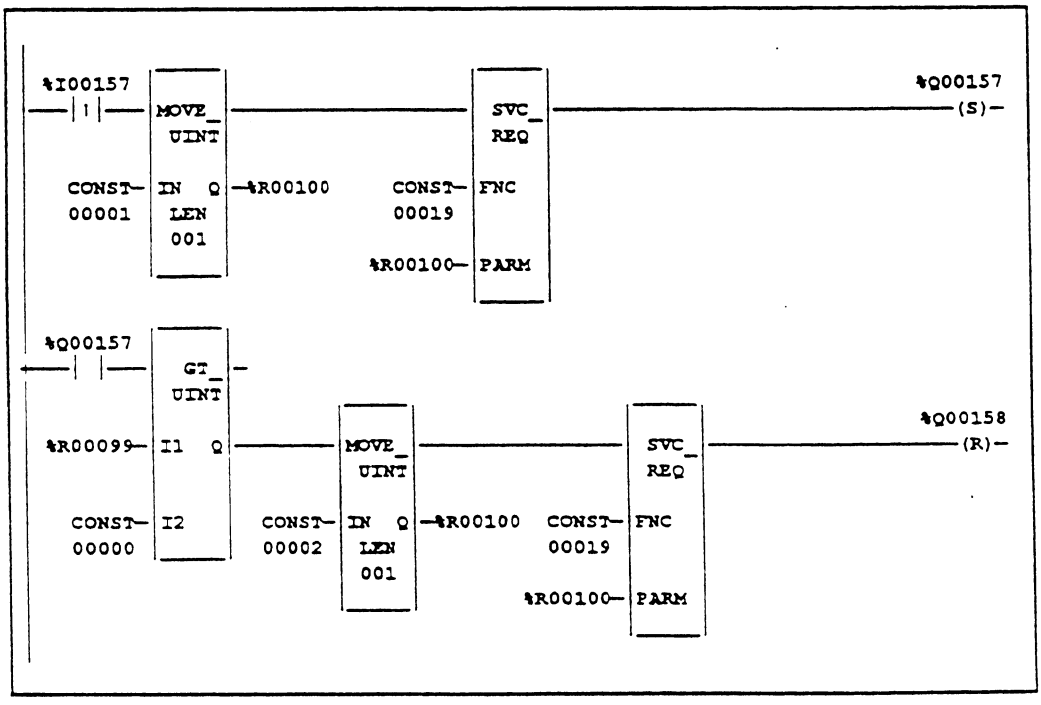
Die übergebenen Parameter zeigen an, welche Funktion ausgeführt werden soll. Der OK-Ausgang wird durchgeschaltet, wenn die Funktion erfolgreich ausgeführt wurde. Er wird abgeschaltet, wenn es sich bei der angeforderten Operation nicht um "RUN freigeben" (1) oder "RUN sperren" (2) handelt.

Der Parameterblock besitzt nur Eingangsparameter und hat das folgende Format:

Adresse	1 = RUN sperren 2 = RUN freigeben

Beispiel:

Im nachstehenden Beispiel wird der Modus "RUN sperren" gesetzt, wenn der Eingang %I00157 eingeschaltet wird. Spule %Q00157 wird durchgeschaltet, wenn die SVCREQ-Funktion fehlerfrei ausgeführt wird. Ist %Q00157 durchgeschaltet und ist der Wert des Registers %R00099 größer als Null, dann wird der Modus umgeschaltet in "RUN freigeben". Wird die SVCREQ-Funktion fehlerfrei ausgeführt, dann wird die Spule %Q00157 abgeschaltet.



SVCREQ #20 Fehlertabellen lesen

Mit der SVCREQ-Funktion 20 können Sie die gesamte SPS- oder E/A-Fehlertabelle abrufen und im Kontaktplanprogramm in vorgegebenen Registern ablegen. Der erste Eingangsparameter gibt an, welche Tabelle gelesen werden soll. Ein zweiter Eingangsparameter (derzeit immer Null) ist für eine zukünftige Erweiterung vorgesehen, mit der ein bestimmter Fehlereintrag gelesen werden kann. Die Daten aus der Fehlertabelle werden in den Parameterblock nach den Eingangsparametern abgelegt.

Der OK-Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde. Er ist abgeschaltet, wenn es sich bei der angeforderten Operation nicht um "SPS-Fehlertabelle lesen" (0) oder "E/A-Fehlertabelle lesen" (1) handelt. Ist die angegebene Fehlertabelle leer, dann schaltet die Funktion zwar den OK-Ausgang durch, gibt jedoch keine Daten zurück.

Parameterblock:

Der Parameterblock umfaßt Ein- und Ausgabeparameter. Das Eingabeformat ist:

Adresse	0 = SPS-Fehlertabelle lesen 1 = E/A-Fehlertabelle lesen
Adresse +1	Immer Null (0)

Das Ausgabeformat ist:

Adresse	0 = SPS-Fehlertabelle lesen 1 = E/A-Fehlertabelle lesen
Adresse +1	Immer Null (0)
Adresse + 2 bis Adresse +14	Nicht belegt
Adresse +15 Adresse +16 Adresse +17	Zeit seit dem letzten Löschen
Adresse +18	Anzahl Fehler seit dem letzten Löschen
Adresse +19	Anzahl Fehler in Warteschlange
Adresse +20	Anzahl gelesener Fehler
Adresse +21	Beginn der Fehlerdaten

Jeder Eintrag in der Fehlertabelle ist 21 Worte (42 Bytes) lang. Maximal gibt es 16 Einträge in der SPS-Fehlertabelle und 32 Einträge in der E/A-Fehlertabelle. Ist die Fehlertabelle leer, dann werden keine Daten zurückgegeben.

Die nachstehende Tabelle zeigt, in welchem Format die Daten von SPS-Fehlertabelle bzw. E/A-Fehlertabelle zurückgegeben werden.

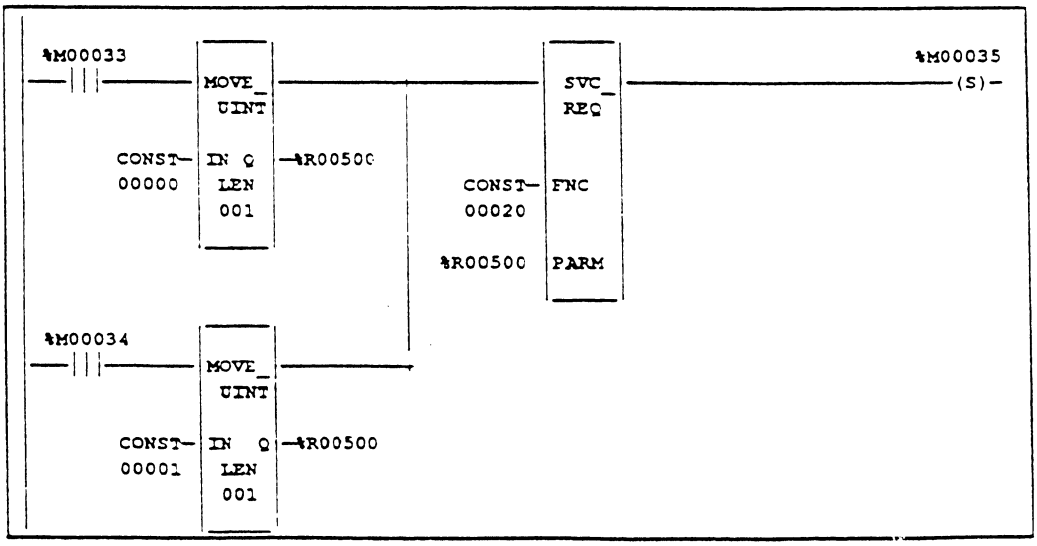
Adresse	SPS-Fehlertabelle	E/A-Fehlertabelle
Adresse +21	Lang/kurz	Lang/kurz
Adresse +22	Reserve	Referenzadresse
Adresse +23	SPS-Fehleradresse	E/A-Fehleradresse
Adresse +24	SPS-Fehleradresse	E/A-Fehleradresse
Adresse +25	Fehlergruppe und Maßnahme	E/A-Fehleradresse
Adresse +26	Fehlercode	Fehlergruppe und Maßnahme
Adresse +27	Zusätzliche Fehlerdaten	Fehlerkategorie und Typ
Adresse +29	Zusätzliche Fehlerdaten	Fehlerbeschreibung
Adresse +29	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +30	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +31	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +32	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +33	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +34	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +35	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +36	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +37	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +38	Zusätzliche Fehlerdaten	Fehlerspezifische Daten
Adresse +39	Zeitmarke	Zeitmarke
Adresse +40	Zeitmarke	Zeitmarke
Adresse +41	Zeitmarke	Zeitmarke

Die Anzeige "Lang/kurz" im ersten Byte von Adresse +21 gibt die Menge der Fehlerdaten in dem Fehlereintrag an. Bei der SPS-Fehlertabelle gibt der Wert 00 in diesem Feld an, daß der Fehlereintrag 8 Bytes zusätzliche Fehlerdaten enthält, der Wert 01 zeigt dagegen 24 Bytes zusätzliche Fehlerdaten an. Bei der E/A-Fehlertabelle steht 02 für 5 Bytes fehlerspezifische Daten und 03 für 21 Bytes.

Eine Erläuterung der einzelnen Felder finden Sie in GFK-0262.

Beispiel:

Wird im nachstehenden Beispiel Eingang %M00033 eingeschaltet, dann wird die SPS-Fehlertabelle gelesen. Wird %M00034 eingeschaltet, dann wird die E/A-Fehlertabelle gelesen. Spule %M00035 wird durchgeschaltet, wenn die SVCREQ-Funktion fehlerfrei durchgeführt wurde.



PID

Die PID-Funktion löst bei jeder Ausführung eine Regelgleichung. Die Funktionsblockdaten belegen 40 Register in einer Regeldatentabelle. Die ersten 35 Register sind für die Funktion reserviert und dürfen vom Anwenderprogramm nicht verwendet werden. Die letzten 5 Register sind für externe Verwendung reserviert.

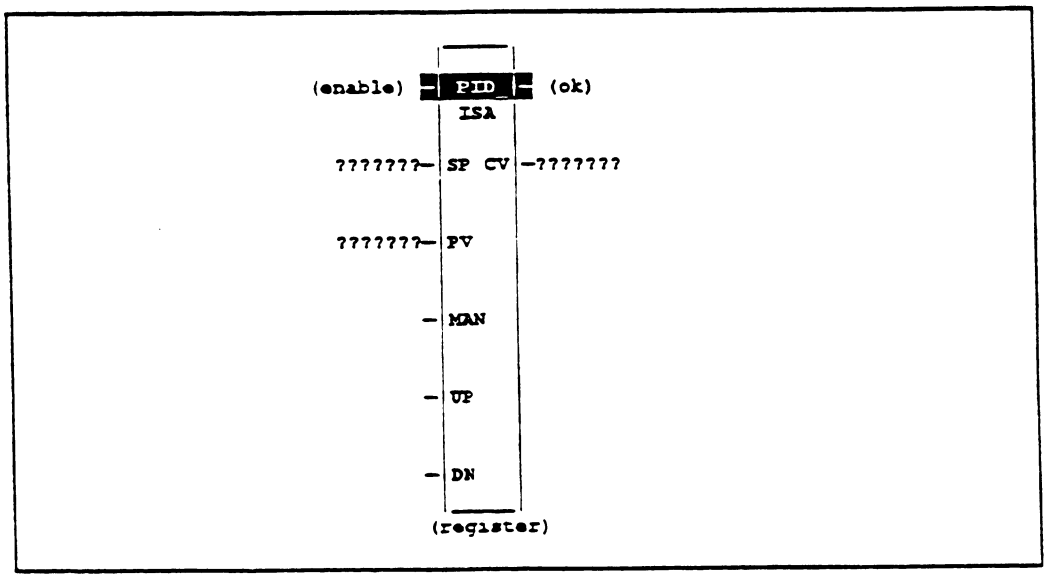
Register können nicht gemeinsam benutzt werden. Wird die gleiche PID-Funktion an unterschiedlichen Stellen zur Steuerung mehrerer Regelkreise mehrfach benutzt, dann muß bei jedem Auftreten ein eigener Block von 40 Registern zur Verfügung stehen.

Die Funktionen PIDISA und PIDIND bilden zwei PID-Regelalgorithmen (Proportional/Integral/Differential).

Die PID-Funktion besitzt sieben Eingänge: einen Boolschen Freigabeeingang (EN), einen Prozeß-Sollwert (SP), eine Prozeßvariable (PV), einen Boolschen Umschalter Hand/Automatik (MAN), eine manuelle Aufwärtsverstellung (UP) und eine manuelle Abwärtsverstellung (DN). Darüberhinaus besitzt die Funktion zwei Ausgangsparameter, über die die erfolgreiche Durchführung (OK) angezeigt bzw. die Regelgröße (CV) ausgegeben wird, und eine Adresse, die die Lage eines Parameterblocks angibt, der mit der Funktion verbunden ist.

Liegt Stromfluß an EN an und kein Stromfluß an MAN, dann wird der PID-Algorithmus auf SP und PV angewandt, und das Ergebnis in CV eingetragen. Wird die PID-Funktion erfolgreich ausgeführt, dann wird OK auf "wahr" gesetzt; im anderen Fall wird OK "falsch".

Liegt Stromfluß an EN und MAN an, dann wird der PID-Block auf Handbetrieb eingestellt. Der CV-Ausgang hält seinen momentanen Wert und kann über die Eingänge UP und DN verstellt werden. Während der PID-Block im Handbetrieb ist, wird der PID-Algorithmus berechnet, so daß das berechnete Ergebnis mit dem von Hand geregelten CV-Wert Schritt hält. Dies verhindert, daß die PID-Funktion im Handbetrieb einen Integralanteil aufbaut, und ermöglicht einen stoßfreien Übergang zurück in Automatikbetrieb.



Parameter:

Parameter	Beschreibung
enable	Die PID-Funktion wird durchgeführt, wenn dieser Eingang aktiviert wird.
SP	SP ist der Regelsollwert.
PV	PV ist die geregelte Prozeßvariable.
MAN	Ist MAN aktiv, dann ist die PID-Funktion in Handbetrieb
UP	Der CV-Ausgang wird nach oben verändert, wenn dieser Eingang in Handbetrieb aktiviert wird.
DN	Der CV-Ausgang wird nach unten verändert, wenn dieser Eingang in Handbetrieb aktiviert wird.
Address	Die Adresse der PID-Steuerblockinformation.
ok	Dieser Ausgang wird durchgeschaltet, wenn die Funktion fehlerfrei ausgeführt wurde.
CV	CV ist der Ausgang der Regelgröße.

Zulässige Speichertypen:

Parameter	flow	%I	%Q	%M	%T	%S	%G	%R	%P	%L	%AI	%AQ	const	none
enable	•													
SP	•	•	•	•	•		•	•	•	•	•	•	•	
PV	•	•	•	•	•		•	•	•	•	•	•		
MAN	•													•
UP	•													•
DN	•													•
address								•	•	•				
ok	•													•
CV	•	•	•	•	•		•	•	•	•	•	•		

• = Zulässiger Datentyp oder Platz, an dem Strom durch die Funktion fließen kann.

PID-Funktionsblockdaten:

Der PID-Funktionsblock enthält die in der Tabelle auf der nächsten Seite aufgelisteten Daten. %adr ist die Anfangsadresse, die dem PID-Funktionsblock im Adreßfeld zugewiesen wurde.

%adr+0000	Kreisnummer **
%adr+0001	Algorithmus **
%adr+0002	Ausführungsintervall *
%adr+0003	Totzone + *
%adr+0004	Totzone - *
%adr+0005	P-Verstärkung *
%adr+0006	D-Verstärkung *
%adr+0007	I-Zeit *
%adr+0008	Verzerrung *
%adr+0009	Obere Verriegelung *
%adr+0010	Untere Verriegelung *
%adr+0011	Nachführgrenze *
%adr+0012	Konfigurationswort *
%adr+0013	Handbefehl *
%adr+0014	Steuerwort **
%adr+0015	SP intern **
%adr+0016	CV intern **
%adr+0017	PV intern **
%adr+0018	Ausgangswert **
%adr+0019	Speicherung D-Größe **
%adr+0020	Speicherung I-Größe **
%adr+0021	Speicherung I-Größe **
%adr+0022	Speicherung Änderungsgröße
%adr+0023	Takt
%adr+0024	
%adr+0025	(letzte Ausführungszeit)
%adr+0026	Y Rest Speicherung
%adr+0027	Unterer Bereich von SP, PV *
%adr+0028	Oberer Bereich von SP, PV *
%adr+0029-%adr+0034	Reserviert für internen Gebrauch
%adr+0029-%adr+0034	Reserviert für internen Gebrauch
%adr+0035-%adr+0039	Reserviert für internen Gebrauch

* Vom Anwender einstellbar

** Von SPS eingestellt und verwaltet

Kreisnummer, Ausführungsintervall, Totzone +/-, P- und D-Verstärkung, I-Zeit, Verzerrung, obere/untere Verriegelung, Nachführgrenze und Konfigurationswort sind Werte, die vom Anwenderprogramm eingestellt werden müssen. Die restlichen Werte werden vom PID-Funktionsblock verwaltet.

Tabelle 11-2. PID-Parameterblock

Parameter	Beschreibung
Kreisnummer	Die Kreisnummer, eine ganze Zahl ohne Vorzeichen, bildet eine gemeinsame Identifizierung in der SPS, bei der die Kreisnummer über eine Bedienschnittstelle definiert wird. Die Kreisnummer wird unterhalb der Blockadresse dargestellt, wenn das Programm mit der Logicmaster 90-70 Software überwacht wird. Die Verwendung der Kreisnummer erfolgt wahlweise.
Algorithmus	Dieser Wert ist eine vorzeichenlose ganze Zahl, die von der SPS gesetzt wird und angibt, welcher Algorithmus von diesem Kreis verwendet wird. Der ISA-Algorithmus wird mit Algorithmus 1 bezeichnet, der unabhängige Algorithmus mit Algorithmus 2.
Ausführungsintervall	Mit Ausführungsintervall wird in Schritten von 0,01 Sekunden die Zeit angegeben, die zwischen den Ausführungen des Funktionsblocks liegt. Die PID-Funktion wird mit diesem Intervall berechnet. Die Funktion gleicht die seit der letzten Ausführung verstrichene Zeit mit einer Genauigkeit von 100 Mikrosekunden aus. Wird dieser Wert auf 0 gesetzt, dann wird die Funktion jedesmal ausgeführt, wenn sie freigegeben wird.
Totzone (+/-)	Vorzeichenbehaftete ganze Zahlen legen die obere und untere Grenze der Totzone in Zählwerten fest. Wird keine Totzone benötigt, dann sollte dieser Wert auf 0 gesetzt werden. Liegt der Fehler zwischen den Werten (+) und (-) der Totzone, dann wird die Funktion mit dem Fehlerwert 0 gelöst. Das heißt, daß der Fehler über diese Grenzen hinweg wachsen muß, ehe der PID-Block reagiert und den CV-Ausgang ausregelt.
P-Verstärkung	Ein vorzeichenbehafteter ganzzahliger Wert, mit dem die Proportionalverstärkung in Hundertstelsekunden eingestellt wird.
D-Verstärkung	Ein vorzeichenbehafteter ganzzahliger Wert, mit dem die Differentialverstärkung in Hundertstelsekunden eingestellt wird.
I-Zeit	Ein vorzeichenbehafteter ganzzahliger Wert, mit dem die Integrationszeit in Wiederholungen pro tausend Sekunden eingestellt wird.
Verzerrung	Ein vorzeichenbehafteter ganzzahliger Wert, mit dem die Verzerrung in Zählwerten eingestellt wird. Durch Einstellen dieses Wertes kann Vorwärtsregelung realisiert werden.
Obere und untere Verriegelung	Vorzeichenbehaftete ganzzahlige Werte, mit denen die oberen und unteren Grenzwerte des CV-Ausgangssignals in Zählwerten festgelegt werden. Wird eine Verriegelungsgrenze erreicht, dann wird die Integralgröße auf einen Wert eingestellt, der den Ausgang auf dem Verriegelungswert festhält.
Nachführgrenze	Ein ganzzahliger Wert ohne Vorzeichen, der die Grenze der Nachführgeschwindigkeit am Ausgang für positive und negative Anderungen angibt. Mit diesem Wert wird festgelegt, wie schnell der Ausgangswert von 0 auf 100% gehen kann. Hierdurch wird die Anderungsgeschwindigkeit der Integralgröße begrenzt, wodurch Schwingungen verhindert werden. Wird keine Grenze der Nachführgeschwindigkeit gewünscht, dann muß diese Größe auf Null gesetzt werden. Die Grenze der Nachführgeschwindigkeit wird in Sekunden für Vollausschlag angegeben.

Tabelle 11-2. PID-Parameterblock (Fortsetzung)

Parameter	Beschreibung
Konfigurationswort	Ein Wortwert mit folgendem Format: **0 = Fehlergröße.** Ist dieses Bit auf 0 gesetzt, dann ist der Fehlergröße SP-PV. Ist das Bit 1, dann ist er PV-SP. **1 = Ausgangspolarität.** Ist dieses Bit 0, dann wird der CV-Ausgang auf den Ausgangswert der PID-Berechnung eingestellt. Ist dieses Bit 1, dann wird der CV-Ausgang auf den negativen Ausgangswert der PID-Berechnung eingestellt. **2 = Differentialwirkung auf PV.** Ist dieses Bit 0, dann wirkt das Differentialverhalten auf die Fehlergröße. Ist dieses Bit 1, dann wirkt das Differentialverhalten auf PV. Alle anderen Bits sollten auf Null gesetzt werden.
Handbefehl	Ein vorzeichenbehafteter Wortwert, der im Handbetrieb den Ausgangswert angibt.
Steuerwort	Eine diskrete Datenstruktur mit dem folgenden Format: 0 = OVERRIDE 1 = Automatik/Manuell 2 = Freigeben 3 = Höher 4 = Niedriger **OVERRIDE:** Wurde das OVERRIDE-Bit auf 1 gesetzt, dann wird der Funktionsblock auf der Grundlage der aktuellen Werte von "Höher", "Niedriger" und "Manuell" ausgeführt; diese Werte werden nicht mit den diskreten Eingängen in den Funktionsblock eingetragen. Steht das OVERRIDE-Bit auf 0, dann werden die Werte von "Höher", "Niedriger" und "Manuell" auf die von den diskreten Eingängen des Funktionsblocks vorgegebenen Werte eingestellt. Das OVERRIDE-Bit beeinflußt auch den für SP verwendeten Wert. Wurde es gesetzt, dann aktualisiert der Funktionsblock den Wert für SP nicht und arbeitet auf der Basis des SP-Wertes aus der Datenstruktur. Durch das OVERRIDE-Bit kann der Bediener über ein Hand-Bediengerät die Boolschen Eingänge zum Funktionsblock steuern. Da SP nicht aktualisiert wird, kann der Bediener über das Bediengerät Uberschreibungen setzen und den Sollwert steuern. **Freigeben:** Das Freigabebit verfolgt den Freigabeeingang zum Funktionsblock. **Manuell/Höher/Niedriger:** Diese drei Bits repräsentieren die Zustände der drei Boolschen Eingänge zum Funktionsblock, wenn das OVERRIDE-Bit auf 0 gesetzt ist. Im anderen Fall können sie durch eine externe Quelle manipuliert werden.
SP	Dieser vorzeichenbehaftete Wortwert stellt den Sollwerteingang in den Funktionsblock dar.
CV	Dieser vorzeichenbehaftete Wortwert stellt den CV-Ausgang des Funktionsblocks dar.
PV	Dieser vorzeichenbehaftete Wortwert stellt den Prozeßvariableneingang des Funktionsblocks dar.

Tabelle 11-2. PID-Parameterblock (Fortsetzung)

Parameter	Beschreibung
Ausgangswert	Dieser vorzeichenbehaftete Wortwert stellt den Ausgangswert des Funktionsblocks vor der wahlweisen Invertierung dar. Wurde keine Ausgangswertinvertierung konfiguriert und ist das Ausgangspolaritätsbit im Steuerwort auf 0 gesetzt, dann ist dieser Wert gleich dem CV-Ausgangswert. Wurde Ausgangswertinvertierung konfiguriert und ist das Ausgangspolaritätsbit im Steuerwort auf 1 gesetzt, dann ist dieser Wert gleich dem negativen CV-Ausgangswert.
Speicherung D-Größe	Wird intern zur Speicherung von Zwischenwerten verwendet. In diese Adresse darf nicht geschrieben werden.
Speicherung I-Größe	Wird intern zur Speicherung von Zwischenwerten verwendet. In diese Adresse darf nicht geschrieben werden.
Speicherung Anderungsgröße	Wird intern zur Speicherung von Zwischenwerten verwendet. In diese Adresse darf nicht geschrieben werden.
Takt	Interne Speicherung der Betriebszeit (letzte Ausführungszeit). In diese Adresse darf nicht geschrieben werden.
Unterer Bereich	Unterer Bereich von SP, PV für Frontplattenanzeige.
Oberer Bereich	Oberer Bereich von SP, PV für Frontplattenanzeige.
Reserviert	Reserviert für die Verwendung durch GE Fanuc. Darf nicht für andere Zwecke verwendet werden.

Initialisierungswerte

In der folgenden Tabelle sind die Initialisierungswerte für den PID-Funktionsblock zusammengestellt.

Register	Zweck	Einheiten	Einstellvorschlag	Bereich
%adr+0	Kreisnummer		1	
%adr+2	Ausführungsintervall	10 ms	100 ms (10)	0 bis 10,9 min
%adr+3	Totzone +	Zählwerte	320	0 bis 100% vom Fehler
%adr+4	Totzone -	Zählwerte	320	0 bis -100% vom Fehler
%adr+5	P-Verstärkung	0,01 %/%	Anwendereinstellung	0 bis 327,67 %/%
%adr+6	D-Verstärkung	0,01 s	Anwendereinstellung	0 bis 327,67 s
%adr+7	I-Zeit	Wiederholungen pro 1000 s	Anwendereinstellung	0 bis 32,767 Wiederh./s
%adr+8	Verzerrung	Zählwerte	50% (16000)	-100% bis +100%
%adr+9	Obere Verriegelung	Zählwerte	100% (32000)	-100% bis +100%
%adr+10	Untere Verriegelung	Zählwerte	0% (0)	-100% bis +100%
%adr+11	Nachführgrenze	Sekunden pro Vollausschlag	0	0 bis 32.767

Funktionsbeschreibung

Nachdem der PID-Funktionsblock freigegeben wurde, wird das konfigurierte Ausführungsintervall (%adr+2) mit der Zeit verglichen, die seit der letzen Ausführung des Funktionsblockes verstrichen ist. Der Funktionsblock wird bearbeitet, wenn die entsprechende Zeit verstrichen ist. Die PID-Regelgleichung wird auf der Grundlage der seit der letzten Ausführung tatsächlich verstrichenen Zeit gelöst, nicht nach dem programmierten Ausführungsintervall.

Liegt die berechnete Regelgröße außerhalb der konfigurierten oberen oder unteren Verriegelung (%adr+9 bzw. %adr+10) oder hat sie sich um einen Wert geändert, der größer als die Nachführgrenze (%adr+11) ist, dann wird die Regelgröße auf dem entsprechenden Grenzwert gehalten und die Speicherung der I-Größe wird entsprechend angepaßt.

Nachdem die Regelgröße berechnet wurde, wird sie im Handbefehl-Register (%adr+13) und, wenn die Regelung im Automatikmodus abläuft, im Regelgrößen-Speicherregister (%adr+16) abgelegt. Ist der Funktionsblock in Handbetrieb (der Stromfluß erfolgt zum Handeingang), dann wird der Ausgang der Regelgröße auf dem Wert des Handbefehl-Registers gehalten, das dann über die Auf- und Abwärts-eingänge des Funktionsblocks inkrementiert oder dekrementiert werden kann. Das Handbefehl-Register kann auch im Handbetrieb programmgesteuert geladen werden.

Da die gespeicherte I-Größe im Handbetrieb wie beim Erreichen einer Verriegelungsgrenze nachgeführt wird, ist ein stoßfreier Übergang zwischen Hand- und Automatikbetreib gewährleistet. In Handbetrieb ist die Regelgröße auch durch die konfigurierten Verrieglungs- und Nachführgrenzen eingeschränkt. Die Nachführgrenze hält den Bediener davon ab, die Regelgröße im Handbetrieb zu schnell zu verändern.

Unterschied zwischen den Blöcken PIDISA und PIDIND

Wie in nachstehendem Blockschaltbild gezeigt, wirkt beim Standard-ISA-PID-Algorithmus die Proportionalverstärkung auf die P-, D- und I-Größe.

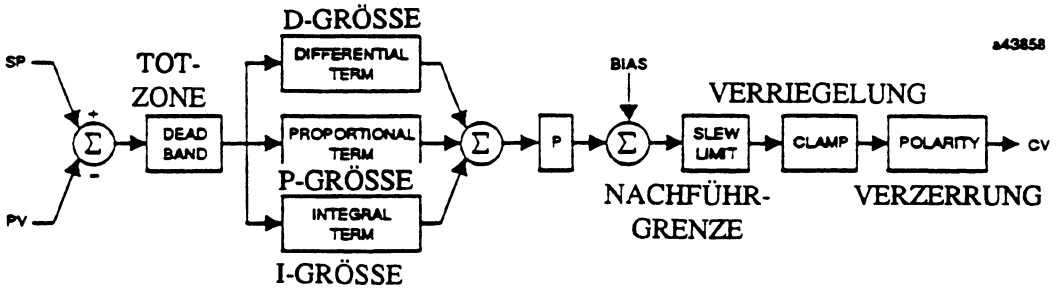


Abbildung 11-1. Standard-ISA-PID-Algorithmus (PIDISA)

Wie in nachstehendem Blockschaltbild gezeigt, wirkt beim unabhängigen Algorithmus die Proportionalverstärkung nur auf die P-Größe. In den übrigen Punkten sind die beiden Algorithmen gleich.

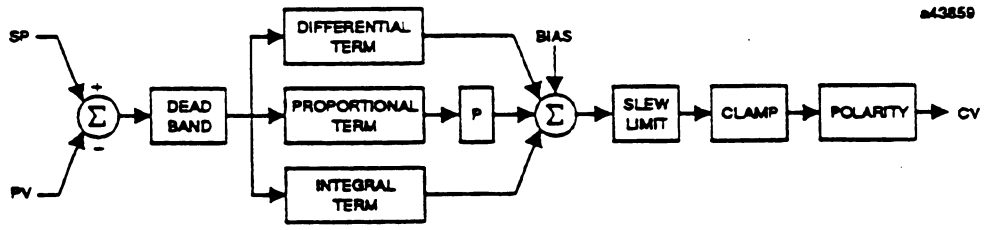
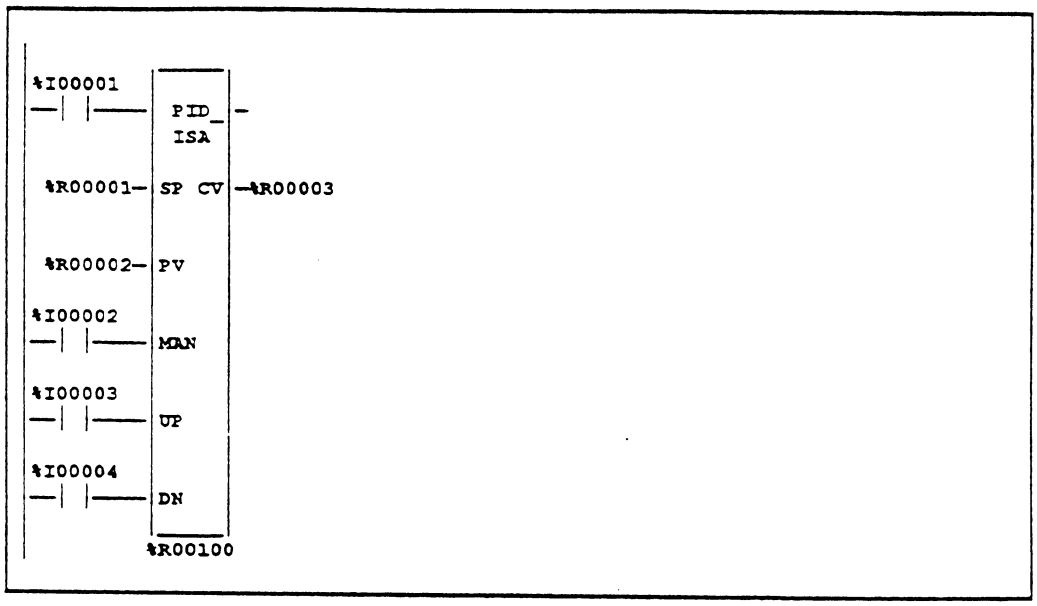


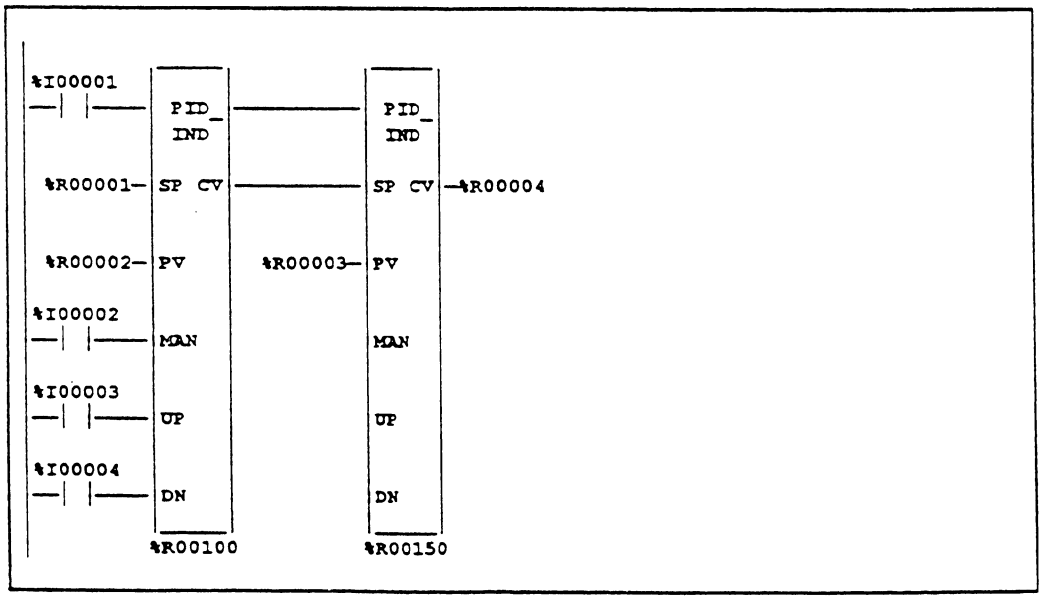
Abbildung 11-2. Unabhängiger PID-Algorithmus (PIDIND)

Beispiel 1:

Im folgenden Beispiel wird die PID-Funktion in einem Strompfad verwendet. %R00001 enthält den Sollwert, %R00002 die Prozeßvariable. %R00100 ist das erste Register im Parameterblock. Jedesmal, wenn %I00001 wahr und %I00002 falsch ist, wird der ISA-PID-Algorithmus auf die Funktionseingänge angewandt und das Ergebnis wird in %R00003 abgelegt. Jedesmal, wenn %I00001 und %I00002 wahr sind, wird das in CV abgelegte Ergebnis von den Zuständen von %I00003 und %I00004 korrigiert.



Im folgenden Beispiel werden zwei PID-Kreise kaskadiert. Der Ausgang des ersten oder äußeren Regelkreises steuert den Sollwerteingang des zweiten, inneren Regelkreises.



Abgleich nach Ziegler und Nichols

Änderungen der Proportional- und Integralverstärkung betreffen die Ausgangsgröße unmittelbar. Diese Parameter sollten daher langsam und in kleinen Schritten verstellt werden, damit das System sich entsprechend einstellen kann. Der Abgleich eines Regelkreises sollte gemäß einer einschlägig bewährten Methode erfolgen. Eine dieser Abgleichmethoden ist das hier beschriebene Verfahren nach Ziegler und Nichols.

1. Stellen Sie die Prozeßverstärkung fest. Geben Sie einen Einheitsschritt auf den Ausgang der Regelgröße und messen Sie nach der Stabilisierung die Antwort der Regelgröße. Diese Antwort ist K, die Prozeßverstärkung.

2. Bestimmen Sie die Prozeß-Verzögerungszeit. Die Prozeß-Verzögerungszeit t kann als die Zeit angenommen werden, die die Prozeßvariable benötigt, um auf eine Schrittänderung der Regelgröße zu reagieren. Dies ist normalerweise der Punkt, an dem die Prozeßvariable ihre maximale Änderungsgeschwindigkeit erreicht hat.

3. Bestimmen Sie die System-Ersatzzeitkonstante. Die System-Ersatzzeitkonstante T kann aus der Zeit ermittelt werden, die von der Prozeßvariablen benötigt wird, 63% ihres stationären Wertes zu erreichen, nachdem ein Schritt an die Regelgröße angelegt wurde, minus der Prozeß-Verzögerungszeit t.

4. Berechnen Sie die Reaktionsgeschwindigkeit R:

   $R = K / T$

5. Berechnen Sie bei Proportionalregelung die Proportionalverstärkung P:

   $P = 1 / (R * T)$

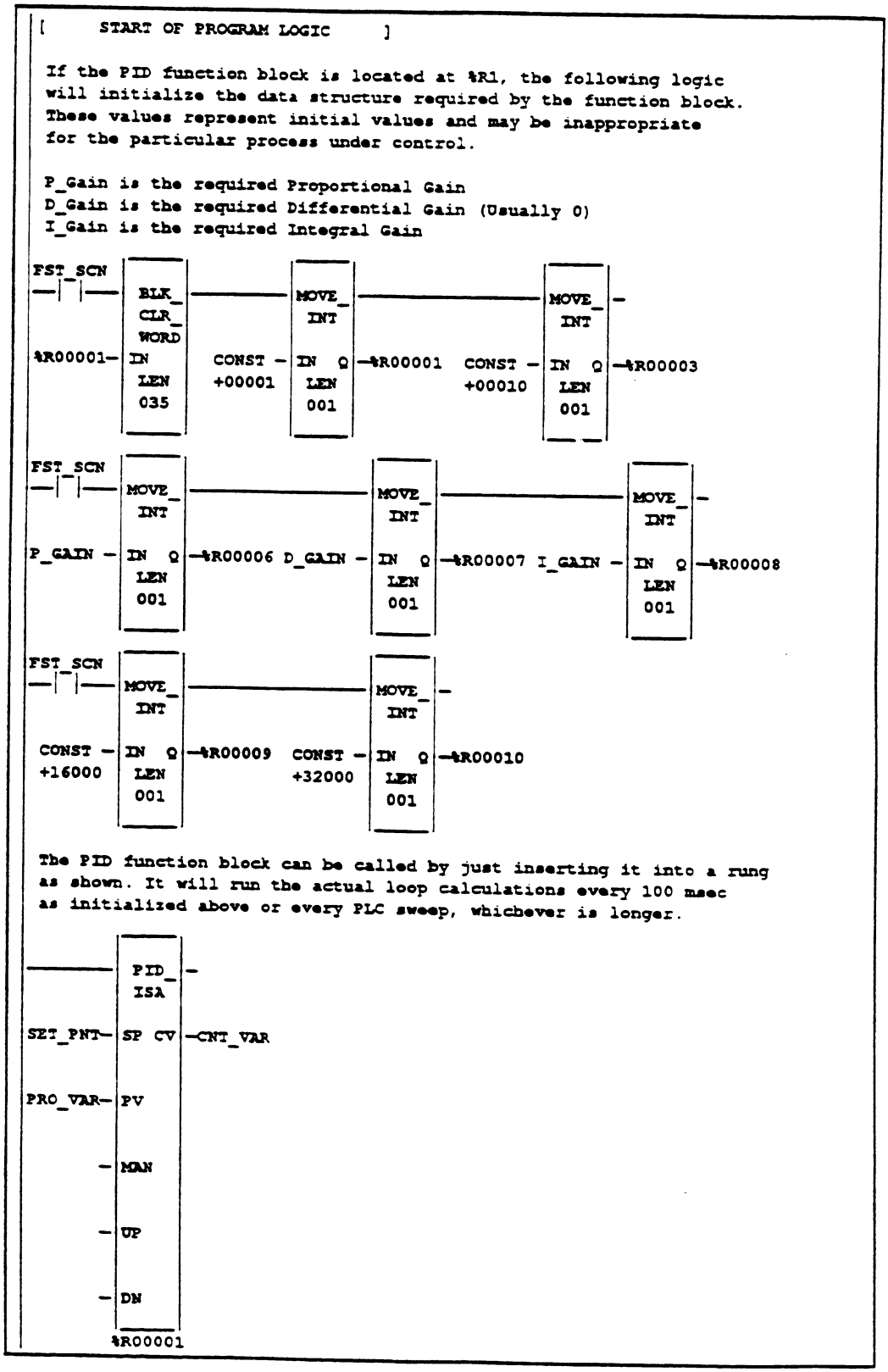
6. Berechnen Sie bei PI-Regelung Proportionalverstärkung P und Integralverstärkung I:

   $P = 0{,}9 / (R * t)$

   $I = 0{,}3 * P / t$

Diese Werte sollten bei dem Abgleichvorgang nur als Anfangswerte verwendet werden. Sie können sich bei zeitvarianten oder nichtlinearen Prozessen mit den Prozeß-Arbeitspunkten verändern. Um die geeigneten Einstellparameter zu erhalten, sollten alle Abgleichvorgänge letzlich von Hand durchgeführt werden und der Prozeß über sämtliche Betriebsbedingungen und Punkte überwacht werden.

Im nachstehenden Beispiel werden Initialisierung und Programmierung des PID-Funktionsblocks dargestellt.



ANHANG A

Dieses Glossar besteht aus zwei Teilen. Der erste Teil ist ein Glossar der bei der SPS Serie 90-70 verwendeten Fachausdrücke, der zweite Teil ist ein Glossar der Basisanweisungen und Referenztypen.

Glossar der Fachausdrücke für die SPS Serie 90-70

Abnehmbarer Klemmenteil - Der von der Vorderseite einer Modulplatine abnehmbare Teil, der die Schraubanschlüsse für die Prozeßverdrahtung enthält.

Abschluß-Widerstandsnetzwerk - Ein Widerstandsnetzwerk, das in einem Abschlußstecker eingebaut ist und mit dem die E/A-Bussignale ordnungsgemäß abgeschlossen werden.

Abschlußstecker - Ein Stecker, der ein Widerstandsnetzwerk enthält, das am Ende des E/A-Busses zum ordnungsgemäßen Abschluß der Bussignale benötigt wird. Bei einer SPS Serie 90-70 muß der Abschlußstecker auf den leeren Steckverbinder des letzten Bus-Receivermoduls der E/A-Buskette aufgesteckt werden.

Adresse - Eine Zahl, die nach einem Referenztyp steht, und mit diesem zusammen eine bestimmte Anwenderreferenz bezeichnet. Beispiel: %Innnnn. Hier ist %I der Referenztyp und nnnnn die Adresse.

Alarmprozessor - Eine Softwarefunktion, die E/A- und Systemfehler mit einem Zeiteintrag versieht und in zwei Tafeln ablegt, die vom Programmiergerät aus aufgerufen oder in einen Hostcomputer oder anderen Coprozessor geladen werden können.

Analog - Ein elektrisches Signal, das durch physikalische Variablen (Kraft, Druck, Temperatur, Durchfluß usw.) aktiviert wurde.

Anschlußbeschaltung - Angaben zum Anschluß der Prozeßgeräte an die Ein- und Ausgangsmodule. Bei jedem E/A-Modul ist die Anschlußbeschaltung auf der Innenseite der schwenkbaren Klemmenabdeckung aufgedruckt.

Anwender-Referenztyp - Eine Referenz, die Daten zugeordnet ist und die angibt, im welchem Speicher der SPS die Daten gespeichert sind. Referenzen können bit- (diskret) oder wortorientiert (Register) sein.

Anwenderprogramm - Ein vom Anwender erstelltes Programm, mit dem eine Maschine oder ein Prozeß (d.h. eine Anwendung) gesteuert werden.

Anwenderspeicher - Der Teil des Systemspeichers (batteriegepufferter CMOS RAM), in dem das Anwenderprogramm und die Daten gespeichert sind.

ASCII - Abkürzung für: American Standard Code for Information Interchange. Ein 8-Bit-Code (7 Bits plus 1 Paritätsbit), der zur Datendarstellung verwendet wird.

Ausgangs-Zykluszeit - Die von der CPU benötigte Zeit, um alle E/A-Controller mit neuen Werten zu versorgen. Bei Modell 30 E/A umfaßt dies auch die Zeit, um die Daten in die einzelnen Module einzutragen.

Ausgangsgerät - Physikalisches Gerät, wie Schütz, Relais, Magnetventil usw., das durch die SPS geschaltet wird.

Ausgangsmodul - Ein E/A-Modul, das logische Signale aus der CPU in Pegelwerte umsetzt, die zur Steuerung von Maschinen oder Prozessen verwendet werden können.

Ausgang - Daten, die von der CPU über ein Modul zur Pegelumsetzung ausgegeben werden, um ein externes Gerät oder einen Prozeß zu steuern.

Batterieanschluß - Ein Steckverbinder an einer Lithiumbatterie, mit dem diese Batterie über einen Stecker auf der Leiterplatte angeschlossen wird.

Baud - Eine Einheit, die die Geschwindigkeit bei der Datenübertragung in Bits pro Sekunde angibt.

Befehlszeile - Die Befehlszeile ist bei der Logicmaster 90-70 Software auf dem Bildschirm die vierte Zeile von oben. In ihr werden die eingegebenen Daten und Befehle angezeigt.

Bezeichnung - Eine zusätzliche Textbeschreibung zu einer Referenz. Eine Bezeichnung kann mit oder ohne symbolischer Adresse verwendet werden. Bezeichnungen werden auch in der Variablentabelle eingetragen.

Bit - Die kleinste Einheit im Speicher, die zum Speichern von Einzelinformationen mit zwei Zuständen verwendet wird (z.B. Null/Eins, Ein/Aus, Gut/Schlecht, Ja/Nein). Daten, die mehr als zwei Zustände erfordern (z.B. numerische Werte zwischen 000 und 999) benötigen mehrere Bits (siehe Wort).

Bus - Ein elektrischer Pfad zum Senden und Empfangen von Daten.

Byte - Eine Gruppe binärer Stellen, die als eine Einheit verarbeitet werden. Bei der SPS Serie 90-70 besteht ein Byte aus acht Bits.

Chassisnummer-Brücken - Eine Gruppe binär codierter Brücken auf der Rückwandplatine, direkt hinter der Stromversorgung, über die die eindeutige Chassisnummer eingestellt wird.

Chassisnummer - Eine eindeutige Zahl zwischen 0 und 7, die einem Chassis als Kennung zugewiesen wird.

CMOS-Speichererweiterungsplatine - Eine Tochterplatine mit batteriegepuffertem CMOS RAM-Speicher, die auf die Hauptplatine von CPU oder PCM als Speicher für Anwenderprogramm und Anwenderdaten aufgesteckt wird.

CONFIG.SYS - Diese Datei enthält bestimmte Befehle, die von DOS beim Starten des Computers überprüft werden. Mit CONFIG.SYS können Sie die Standard-Konfigurationseinstellungen Ihres Systems verändern.

CPU-Betriebsartenschalter - Ein Umschalter mit drei Schalterstellungen, der sich oben an der CPU-Platine befindet und mit dem die Betriebsart der CPU (STOP, RUN/DISABLED, RUN/ENBLED) eingestellt werden kann.

CPU (Zentraleinheit) - Das Zentralgerät bzw. die Zentralsteuerung, die auf der Grundlage eines gespeicherten Anwenderprogramms Anweisungen vom Anwender interpretiert, Entscheidungen fällt und Funktionen ausführt.

Datensicherung - (Backup). Die genaue Kopie eines Programms, die zu Sicherungszwecken vor dem Editieren eines Programms angelegt wird.

Datenspeicher - Anwenderreferenzen innerhalb der Serie 90-70 CPU, auf die vom Anwenderprogramm zur Speicherung von binären oder Registerdaten zugegriffen werden kann.

Datentabelle - Eine aufeinanderfolgende Gruppe von Anwender-Referenzen der gleichen Größe, auf die mit Tabellen-Lese- und Schreibfunktionen zugegriffen werden kann.

Directory - (Verzeichnis) Eine Datei, die die Bezeichnungen und Daten anderer Dateien im Computer enthält.

Diskret - Dieser Begriff umfaßt sowohl physikalisch vorhandene als auch interne E/A, die als 1-Bit-Anwenderreferenz dargestellt werden.

DOS (Disk Operating System) - Eine Gruppe von Dienstprogrammen, die die Struktur für die Systemoperation eines Personalcomputers bilden.

E/A-Fehlertafel - Eine Fehlertafel, in der E/A-Fehler aufgelistet sind, die mit Zeit, Datum und Ort gekennzeichnet wurden.

E/A-Modul - Eine Platine, die die Schnittstelle zwischen Prozeßgeräten und der SPS Serie 90-70 bildet.

E/A-Potentialtrennung - Ein Verfahren, bei dem die Prozeßverdrahtung von den logischen Schaltkreisen elektrisch getrennt wird. Normalerweise wird dies durch Optokoppler erreicht.

E/A (Ein-/Ausgang) - Der Teil der SPS, an den Prozeßgeräte angeschlossen werden und der die CPU elektrisch von Störungen trennt.

Eingangs-Zykluszeit - Die Zeit, die die CPU benötigt, um alle E/A-Controller auf neue Eingangswerte abzufragen. Bei der Modell 30 E/A ist hierin auch die Zeit enthalten, die zum Auslesen des Moduls benötigt wird.

Eingangsmodul - Ein E/A-Modul, das Signale von Prozeßgeräten in logische Pegel umwandelt, die von der CPU verarbeitet werden können.

Erdungslasche - Eine Lasche unten an der Chassis-Stromversorgung und am Montagerahmen der Chassis, die mit Erde verbunden werden muß (über die Wechselspannungsquelle).

Erweiterungschassis-Steckleitung - Ein Kabel, über das die Signale des parallelen E/A-Busses zwischen den Chassis übertragen werden. Die Gesamtlänge aller Steckleitungen zwischen CPU-Chassis und dem letzten Chassis in der Kette darf 15 m nicht übersteigen.

Erweiterungschassis - Ein Chassis, das einem System hinzugefügt wird, wenn für einen Anwendungsfall mehr Module benötigt werden, als im CPU-Chassis eingebaut werden können. Bei einer SPS Serie 90-70 sind bis zu sieben Erweiterungschassis möglich.

Firmware - Eine Reihe von Anweisungen, die in einem ROM (Read Only Memory = Festspeicher) enthalten sind, und die für interne Verarbeitungsfunktionen verwendet werden. Diese Anweisungen bilden die Struktur für Operationen des Anwenderprogramms.

Flansch für Gestellmontage - Flansche an der Vorderseite eines Chassis, mit denen das Chassis in einem 19"-Rahmen eingebaut werden kann.

Flüchtiger Speicher - Ein Speichertyp, bei dem die gespeicherten Daten verlorengehen, wenn die Versorgungsspannung ausfällt. Aus diesem Grund muß dieser Speichertyp mit einer Batterie gepuffert werden. In der SPS Serie 90-70 wird hierzu eine Lithium-Batterie verwendet.

Funktionsblockreferenz - Die Anfangsadresse einer zusammenhängenden Gruppe von Anwenderreferenzen, die von einigen Funktionsblöcken (z.B. Zeitglieder, Zähler oder PID) benötigt wird.

Funktionstaste - Eine Taste (F1 bis F10), deren Funktion von der Software gesteuert wird und die innerhalb des Programms verändert werden kann. Bei der Logicmaster 90 Software werden die aktuellen Tastenbelegungen oben am Bildschirm angezeigt.

Genius-E/A-Block - Ein Modul, das die Schnittstelle zwischen physikalischen Geräten und dem Buscontroller in der SPS Serie 90-70 bildet. Die Blöcke in einem System tauschen mit dem Buscontroller über einen seriellen Bus Daten aus.

Genius-E/A - Ein intelligentes E/A-System, das aus E/A-Blöcken, Buscontrollern und anderen Geräten besteht.

Hardware - Sämtliche mechanischen, elektrischen und elektronischen Geräte, die eine SPS Serie 90-70 und ihre Prozeßumgebung ausmachen.

Haupt-Programmblock - Der Programmblock, der bei allen Anwenderprogrammen vorhanden sein muß. Er enthält die Logik und die %P-Daten. Der Haupt-Programmblock kann bis zu 8 k Worten umfassen.

Hauptchassis - Das Chassis einer SPS Serie 90-70, das die CPU enthält. Dieses Chassis muß immer vorhanden sein und immer als "Chassis 0" definiert werden.

Hauptmenü - Im Hauptmenü von Logicmaster 90 werden alle wichtigen Systemfunktionen zusammen mit den entsprechenden Funktionstasten aufgelistet.

Hexadezimal - Ein Zahlensystem mit der Basis 16, das die Ziffern 0 bis 9 und die Buchstaben A bis F verwendet.

Hilfsmenü - Erläuternde Textmenüs, die aufgerufen werden können, indem die Tasten Alt und K gemeinsam gedrückt werden, und die kontextbezogene Hilfen zu allen Programmier- und Konfigurationsfunktionen enthalten. Mit ALT-I werden Hilfstexte zu mnemonischen Funktionen angezeigt, mit ALT-K erscheint eine Liste aller Sondertasten.

Interrupt-Vereinbarung - Wird zur Verbindung des Interrupteingangs eines Hardwaremoduls mit einem Programmblock verwendet. Die Vereinbarung des Programmblocks, der auf den Eingang reagiert, muß im Programmblock-Vereinbarungsteil des Hauptprogrammblocks stehen. Es können bis zu 64 Interrupt-Vereinbarungen verwendet werden.

Klemmenabdeckung, schwenkbare - Eine Plastikklappe an der Vorderseite der Module, hinter der bestimmte Hardwareelemente des Moduls zugänglich werden.

Klemmenbrücke - Ein U-förmiges Metallstück, mit dem die beiden unteren Klemmen am Stromversorgungs-Klemmenblock gebrückt werden können, um eine Versorgungsspannung von 120 V AC einzustellen. Ist die Brücke nicht eingebaut, dann ist das Netzteil auf 240 V AC eingestellt.

Kodeträger - Ein einmaliger mechanischer Kodeträger aus Kunststoff, der mit den einzelnen Typen von E/A-Modulen mitgeliefert wird und der beim erstmaligen Einbau eines Moduls automatisch in der Mittelschiene der Rückwandplatine einrastet und dort bleibt, selbst wenn das Modul entfernt wird. Dieser Kodeträger bildet eine mechanische Verriegelung, die ein Verwechseln von Modultypen verhindert.

Kommentar - Zusätzlicher erläuternder Text in einem Programm. Es gibt drei verschiedene Kommentartypen: symbolische Adressen, Bezeichnungen und Strompfad-Kommentare.

Konfigurations-Software - Der Anteil der Logicmaster 90 Programmerstellungs-Software, der die Werkzeuge zur Konfiguration der E/A und zahlreicher Systemparameter bereitstellt.

Konstante - Ein fester Wert oder ein unveränderliches Datenelement. Kann in einem Register gespeichert werden.

Kontaktplanprogramm - Die graphische Darstellung kombinatorischer Logik.

K - Abkürzung für Kilo bzw. 1024 in der Computersprache.

Laden - Die Funktion, mit der Programme von der SPS in den Folder des Logicmaster-Systems übertragen werden.

Laufwerk - Das Laufwerk einer Winchester- oder Floppy-Disk-Einheit. Die Kennung eines Laufwerks, z.B. Laufwerk A.

LED-Block - Ein Block, bei dem eine Gruppe LEDs mit vier Spalten zu je acht LEDs oben an jeder diskreten E/A-Platine sowie eine LED unten am Block sitzt. Jede LED in der Gruppe mit vier Spalten zeigt den Zustand des entsprechenden Ein- oder Ausgangs an. Die untere LED mit der Beschriftung FUSE [Sicherung] leuchtet auf, wenn im Modul eine Sicherung durchgebrannt ist.

Lernmodus - Eine Funktion, mit der eine Eingabesequenz für spätere Wiederverwendung gespeichert wird.

Lesen - Abrufen von Daten von der Peripherie oder einem Speichermedium.

Liste - Eine Gruppe aufeinanderfolgender Speicherzellen, die zur Datenbearbeitung verwendet werden. Anfangsadresse und Länge der Liste werden im Anwenderprogramm eingestellt. Auf die Daten wird entweder am Anfang oder am Ende der Liste zugegriffen.

Mikrosekunden (µs) - Eine Millionstel Sekunde. 1×10^{-6} oder 0,000001 Sekunde.

Millisekunden (ms) - Eine Tausendstel Sekunde. 1×10^{-3} oder 0,001 Sekunde.

Mnemonik - Die Abkürzung einer Anweisung, normalerweise ein Akronym aus den Anfangsbuchstaben oder aus Wortteilen.

Modell 30 E/A - Das E/A-Subsystem der SPS Serie 90-70, das aus digitalen, analogen und intelligenten Ein- und Ausgangsmodulen besteht.

Modulabdeckung - Eine Plastikabdeckung auf der Rückseite der CPU- und PCM-Platinen, die die Speicherelemente auf diesen Platinen schützt.

Modul - Eine auswechselbare elektronische Baugruppe, die in Steckverbinder an der Rückwandplatine eingesteckt und gesichert wird, die jedoch im Fehlerfall oder bei Systemänderungen einfach wieder ausgewechselt werden kann. Bei der SPS Serie 90-70 besteht ein Modul aus der Platine und einer Frontplatte sowie (bei E/A-Modulen) aus einem abnehmbaren Klemmenteil.

Monitor-Betriebsart - Eine Betriebsart des Logicmaster 90 Programmiergerätes, bei der das Programmiergerät nur Daten von der SPS abrufen kann, ohne jedoch Daten ändern zu können.

Montageflansche - Flansche an der Rückseite des Chassis, mit denen das Chassis auf einer Montageplatte oder Wand befestigt wird.

Nichtflüchtiger Speicher - Ein Speicher (z.B. PROM), der die in ihm gespeicherte Information erhält, wenn seine Versorgungsspannung abgeschaltet wird.

Nicht nullspannungssichere Spule - Eine Spule, die abschaltet, wenn die Versorgungsspannung weggenommen wird.

Nullspannungssichere Spule - Eine Spule, die bei einem Spannungsausfall in ihrem letzten Zustand verharrt.

ODER (logisch) - Eine logische Operation mit Bits, bei der das Ergebnis 1 ist, wenn ein einziges der verknüpften Bits 1 ist.

Off-Line-Modus - Diese Betriebsart wird für die Programmentwicklung verwendet. Programmiergerät und SPS tauschen keine Daten miteinander aus. Die physikalische Datenverbindung kann dabei intakt sein, das Programmiergerät ist jedoch so eingestellt, daß keine Kommunikation möglich ist. Stromflußanzeige und Referenzwerte werden nicht aktualisiert.

On-Line-Änderungen - Änderungen von E/A- oder Registerreferenzen bzw. wortweise Änderungen, die durchgeführt werden, wenn das Logicmaster 90 Programmiergerät im On-Line-Modus ist und in Programmiergerät und SPS genau das gleiche Programm geladen ist.

On-Line-Modus - Eine Betriebsart, bei der Programmiergerät und SPS Daten in beiden Richtungen miteinander austauschen.

Parallelkommunikation - Eine Methode der Datenübertragung, bei der Daten gleichzeitig über mehrere Leitungen übertragen werden.

Paritätsbit - Ein Bit, das zu einem Wort im Speicher hinzugefügt wird, damit die Quersumme der Bits in einem Wort entweder immer gerade (gerade Parität) oder ungerade (ungerade Parität) ist.

Paritätsfehler - Ein Fehlerzustand, der dann auftritt, wenn eine berechnete Paritätsprüfung (Quersumme) nicht mit dem Paritätsbit übereinstimmt.

Parität - Der erwartete Zustand, entweder gerade oder ungerade, einer Gruppe binärer Werte.

Peripheriegeräte - Externe Geräte, die mit einer SPS Daten austauschen können. Beispiel: Programmiergerät, Drucker usw.

Platte - Eine Hard- oder Floppy-Disk, die zum Speichern und Abrufen von Daten eingesetzt wird.

Programm-Folder (= Ordner) - Ein Unterverzeichnis aller Dateien, die ein Programm ausmachen, einschließlich der zugehörigen Konfigurationsdateien. Der Name eines Programm-Folders kann bis zu sieben Zeichen lang sein.

Programm-Zykluszeit - Die Zeit zwischen zwei aufeinanderfolgenden Anfängen eines Programmzyklus. Der Programmzyklus setzt sich aus folgenden Einzelphasen zusammen: Ausführung der Systemaufgaben bei Zyklusbeginn, Lesen der Eingänge, Ausführung des Anwenderprogramms, Schreiben der Ausgänge, Regenerieren fehlerhafter Module, Abschluß der Minimum-Kontrollsummenberechnung, Einteilen des nächsten Zyklus, Kommunikation mit dem Programmiergerät und anderen intelligenten Modulen, Ausführung von Hintergrundaufgaben.

Programmausführungszeit - Die für die Ausführung aller aktiver Anweisungen im Anwenderprogramm benötigte Zeit.

Programmbezeichnung - Der Name des aktuellen Programms. Meistens ist diese Bezeichnung mit der Bezeichnung des Programm-Folders identisch. Die Programmbezeichnung kann auch bis zu sieben Zeichen lang sein.

Programmblockvereinbarung - Angaben über zusätzliche Logik, die vom Haupt-Programmblock oder anderen Programmblöcken aufgerufen werden kann. Ein Haupt-Programmblock kann bis zu 255 Programmblockvereinbarungen enthalten. Sämtliche Programme, die in einem Programm verwendet werden, müssen im Haupt-Programmblock vereinbart werden.

Programmblock - Eine Einheit in einem Anwenderprogramm, die aus bis zu 8 k Worten Kontaktplanprogramm und 8 k Worten lokalen Registern bestehen kann.

Programmerstellungs-Software - Der Teil der Logicmaster 90 Software, der zur Erstellung von Kontaktplanprogrammen verwendet wird.

Programmiergeräte-Port - Der obere Parallelport des Bus-Transmittermoduls, der über einen 37-poligen Steckverbinder zugänglich ist, und über den das Programmiergerät an die SPS Serie 90-70 angeschlossen werden kann. Die CPU besitzt auch einen eingebauten Port für den seriellen Anschluß eines Programmiergeräts.

Programmiergerät - Das Hardwaregerät, auf dem die Logicmaster 90 Software abläuft. Zum Datenaustausch mit der SPS Serie 90-70 muß das Programmiergerät eine Workstation-Schnittstellenplatine enthalten.

PROM - Programmierbarer Festwertspeicher, der als nullspannungssicherer Speicher im Werk programmiert wird und vom Anwender nicht verändert werden kann. Enthält normalerweise die Software für interne Systemaufgaben.

RAM - Schreib-Lese-Speicher mit wahlfreiem Zugriff. Ein Halbleiterspeicher, in dem einzelne Bits beliebig abgelegt und wieder gelesen werden können. Solange das System mit Spannung versorgt wird, sind in diesem Speicher die Systemdaten, Programmdateien und zugehörige Daten gespeichert. Da ein RAM jedoch flüchtig ist, d.h. seine Daten verliert, wenn die Versorgungsspannung ausfällt, muß er durch eine Batterie über einen Spannungsausfall hinweg gepuffert werden. In der SPS Serie 90-70 wird zu diesem Zweck bei den CPU- und PCM-Modulen eine Lithium-Batterie mit langer Lebensdauer verwendet.

Rauschen - Unerwünschte elektrische Störungen, die normale Signale überlagern und normalerweise einen Hochfrequenzanteil haben.

Referenzbeschreibung - Ein wahlweise verwendeter Text zu einer Anwenderreferenz. Eine Referenzbeschreibung kann mit oder ohne symbolische Adresse verwendet werden. Referenzbeschreibungen werden auch in die Variablenvereinbarungstabelle eingetragen.

Referenztyp - Eine bestimmte Gruppe von Speichertypen in der SPS Serie 90-70. %I spricht z.B. digitale Eingänge und %Q digitale Ausgänge an. Mit dem Symbol % werden Maschinenreferenzen von symbolischen Adressen unterschieden.

Register - Eine Gruppe von 16 aufeinanderfolgenden Bits im Registerspeicher, die mit %R angesprochen werden. Die Register sind numeriert, beginnend mit 0001. Der Registerspeicher wird zur Zwischenspeicherung numerischer Werte und zur Bitmanipulation verwendet.

RESTART-Taste - Eine Drucktaste an der Vorderseite des PCM, mit dem PCM, ADC und GDC neu initialisiert oder "weich" oder "hart" neu gestartet werden können.

RUN-Modus - Eine Betriebsart der SPS Serie 90-70, bei der die CPU das Anwenderprogramm bearbeitet. Im RUN-Modus sind wiederum die Betriebsarten RUN/ OUTPUTS ENABLED [Ausgänge freigegeben] und RUN/OUTPUTS DISABLED [Ausgänge gesperrt] möglich. Bei RUN/OUTPUTS ENABLED werden alle Teile des Programmzyklus bearbeitet. Bei RUN/OUTPUTS DISABLED läuft der gesamte Zyklus normal ab, lediglich die Ausgänge werden in ihrer Voreinstellung festgehalten.

Rückwandplatine - Eine Gruppe von Steckverbindern, die physikalisch auf eine Platine an der Chassisrückseite montiert sind, und in die die Module eingesteckt werden. Die Steckverbinder sind durch eine Leiterplatte miteinander verbunden.

Schreiben - Daten von einem Speichermedium zum anderen übertragen, aufzeichnen oder kopieren, z.B. von CPU auf Platte.

Serielle Datenverbindung - Eine Methode der Datenübertragung, bei der die Bits sequentiell (und nicht simultan wie bei Parallelübertragung) übertragen werden.

Signifikantes Bit - Ein Bit, das zur Genauigkeit einer Zahl beiträgt. Die signifikanten Bits beginnen bei dem höchstwertigen Bit (MSB), das am meisten zum Zahlenwert beiträgt, und enden mit den niedrigstwertigen Bit (LSB), das am wenigsten beiträgt.

Sollwert - Ein numerischer Wert, der in einer Funktion angegeben wird und der einen Endwert für ein Zeitglied oder eine Zähler angibt.

Speichern - Die Übertragung von Programmen vom Folder des Logicmaster-Systems zur CPU.

Speicherprogrammierbare Steuerung (SPS) - Ein mit Halbleiterelementen aufgebautes industrielles Steuerungsgerät, das Signale von anwenderseitigen Gebern wie Sensoren und Schaltern erhält, diese Signale in ein präzises Muster umwandelt, das von einem im Anwenderspeicher abgelegten Anwender-Kontaktplanprogramm festgelegt wird, und Ausgangssignale zur Steuerung von Prozessen oder anwenderseitigen Geräten wie Relais oder Anlasser liefert. Eine speicherprogrammierbare Steuerung wird normalerweise in Kontaktplanlogik programmiert und ist zum Betrieb in Industrieumgebung geeignet.

SPS-Fehlertafel - Eine Fehlertabelle mit SPS-Fehlern, die durch Zeit, Datum und Ort festgelegt werden.

Statuszeilen - Die drei untersten Zeilen am Bildschirm. In der obersten dieser drei Zeilen werden Informationen über SPS und Programmiergerät angezeigt. In der zweiten Zeile wird das aktuelle Programm angegeben und in der dritten Zeile erscheint der Tastaturstatus. Bei einigen Programmiergerätefunktionen werden in der dritten Zeile Zusatzinformationen angezeigt.

STOP-Modus - Ein Zustand der SPS Serie 90-70, in dem die CPU das Anwenderprogramm nicht mehr bearbeitet. Im STOP-Modus sind möglich: STOP/NO IOSCAN und STOP/IOSCAN. Bei STOP/NO IOSCAN kommuniziert die SPS nur mit dem Programmiergerät und anderen Geräten (LAN-Schnittstelle, PCM usw.), regeneriert fehlerbehaftete Module, konfiguriert Module neu und führt Hintergrundaufgaben durch. Alle anderen Teile des Zyklus werden übersprungen. In STOP/IOSCAN kann die CPU der SPS die E/A überwachen. Mit dieser Betriebsart können Sie die E/A überwachen und austesten, ohne daß das Anwenderprogramm abläuft.

Stromfluß - In einem Kontaktplanprogramm stellt der symbolische Stromfluß die logische Bearbeitung der Programmfunktionen dar. Es ist wichtig, bei den einzelnen Funktionen zu wissen, was passiert, wenn sie Strom empfangen und unter welchen Bedingungen der Stromfluß weitergeschaltet wird.

Strompfad-Kommentar - Ein Strompfad-Kommentar besteht aus maximal 2048 Textzeichen.

Strompfad-Kommentar - Ein Strompfad-Kommentar kann aus max. 2048 Zeichen bestehen. Ein Strompfad-Kommentar wird über einen COMMENT-Funktionsblock mit einem bestimmten Strompfad verbunden.

Strompfad - Eine Einheit des Kontaktplanprogramms. Ein Strompfad kann bis zu acht parallele Programmzeilen enthalten.

Stromversorgungskabel - Ein Kabel, mit dem ein zweites Chassis (ohne Stromversorgung) an ein anderes Chassis (mit Stromversorgung) angeschlossen werden kann, wenn zur Versorgung der beiden Chassis eine Stromversorgung ausreichend ist.

Symbolische Adresse - Eine zusätzliche Kennung für eine Maschinenreferenz, die aus bis zu 7 Zeichen bestehen kann. Sämtliche vom Programmblock verwendeten symbolischen Adressen werden in der Variablenvereinbarungstabelle eingetragen.

UND (logisch) - Eine logische Operation, die mit Bits durchgeführt wird. Alle Bits müssen 1 sein, damit das Ergebnis 1 wird.

Variablenvereinbarung - Der Teil eines Programms, mit dem symbolische Adressen und Bezeichnungen von Anwenderreferenzen angelegt, angezeigt und verändert werden. Variablenvereinbarungen können in einer Tabelle angezeigt werden, die bis zu 2000 Einträge erlaubt.

Verbindung - Horizontale und vertikale Verbindungen werden in einem Kontaktplanprogramm dazu verwendet, Strom um ein Element im Programm herumzuführen, oder um Elemente seriell oder parallel miteinander zu verbinden.

Vergleichen - Eine Funktion, mit der der Programminhalt verglichen wird. Das Programm im Systemspeicher kann mit einem Programm von der CPU oder einem Plattenspeicher verglichen werden.

Wort - Ein Maß für die Speichergröße. Normalerweise ist ein Wort 4, 8 oder 16 Bits lang. Bei der SPS Serie 90-70 beträgt die Wortlänge 16 Bits.

Zähler - Ein Funktionsblock, der so programmiert werden kann, daß er eine vorgegebene Anzahl von EIN/AUS-Übergängen aufzeichnet.

Zeitglied - Ein Funktionsblock, mit dem der Operationszyklus anderer Geräte durch ein Soll- und Ist-Zeitintervall gesteuert werden kann.

Zykluszeitüberwachung (Watchdog) - Ein Zeitglied in der CPU, mit dem überwacht wird, ob bestimmte Hardwarebedingungen innerhalb einer vorgegebenen Zeit eintreffen. Mögliche Überwachungszeiten, die über das Programmiergerät eingestellt werden können, liegen zwischen 10 ms und 2550 ms. Der voreingestellte Wert bei der Serie 90-70 CPU beträgt 200 ms.

Zyklus - Die wiederholte Ausführung der gesamten Programmlogik, E/A-Aktualisierung, Peripherie-Aktualisierung und des Selbsttests. Der Zyklus läuft automatisch mit zahlreichen Wiederholungen pro Sekunde ab.

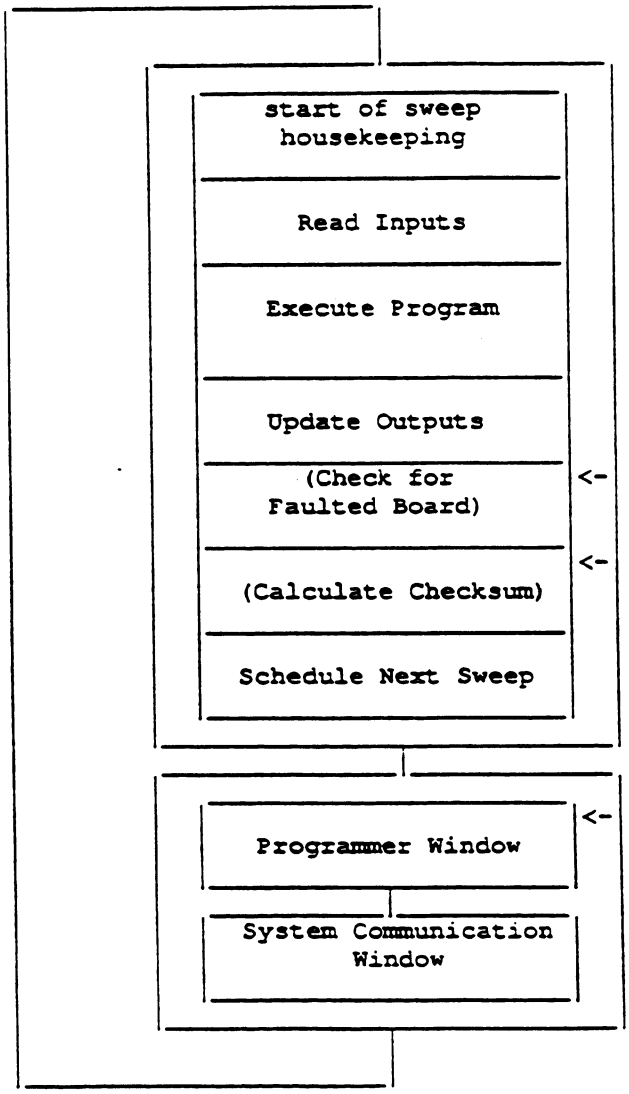
Glossar der Basisanweisungen und Referenztypen der SPS Serie 90-70

Basisanweisung	Bezeichnung	Oberbegriff
--] [--	Schließerkontakt	Kontakt
--]/[--	Offnerkontakt	Kontakt
--]↑[--	Kontakt für positiven Ubergang	Kontakt
--]↓[--	Kontakt für negativen Ubergang	Kontakt
--[FAULT]--	Kontakt "Fehler"	Kontakt
--[NOFLT]--	Kontakt "kein Fehler"	Kontakt
--[HIALR]--	Kontakt "oberer Grenzwert"	Kontakt
--[LOALR]--	Kontakt "unterer Grenzwert"	Kontakt
--( )--	Merker, Spule	Spule
--(/)--	Negierter Merker	Spule
--(S)--	Setz-Merker	Spule
--(R)--	Rücksetz-Merker	Spule
--(↑)--	Merker für positive Anderung	Spule
--(↓)--	Merker für negative Anderung	Spule
--(M)--	Haftmerker	Spule
--(/M)--	Negierter Haftmerker	Spule
--(SM)--	Haftmerker, Setzkreis	Spule
-(RM)-	Haftmerker, Rücksetzkreis	Spule
-------	Horizontalverbindung	Verbindung
|	Vertikalverbindung	Verbindung

Referenztyp	Bezeichnung	Oberbegriff
%I	Eingang	Diskret
%Q	Ausgang	Diskret
%M	Intern	Diskret
%T	Temporär	Diskret
%G	Global	Diskret
%S	System	Diskret
%SA	System	Diskret
%SB	System	Diskret
%SC	System	Diskret
%R	Register	Register
%P	Programmregister	Register
%L	Lokales Register	Register
%AI	Analog-Eingangsregister	Register
%AQ	Analog-Ausgangsregister	Register
%Rnnnnn	nnnnn ist die Adresse	

ANHANG B

Die CPU führt ein Anwenderprogramm aus, aktualisiert die E/A, und führt Datenaustausch und andere Aufgaben durch. Diese Aufgaben werden in einem wiederholten Ablauf durchgeführt, dem CPU-Zyklus, der in der nachstehenden Abbildung dargestellt ist.



← Dieser Zyklusteil wird im Standardmodus oder bei konstanter Zyklusdauer immer ausgeführt.

← Dies tritt auf, wenn ein Modulfehler erkannt wird.

← Wird nur deshalb hier durchgeführt, weil es nicht oben gemacht wurde.

← Wird nicht durchgeführt, wenn kein Programmiergerät angeschlossen ist.

Unter bestimmten Umständen werden manche Zyklusteile nicht ausgeführt. Zum Beispiel:

- Ist die CPU in der Betriebsart mit konstanter Zyklusdauer, dann werden am Zyklusende nur einige oder keine Windows ausgeführt.
- Ist die CPU im STOP-Modus, dann werden Programmausführung und E/A-Aktualisierung nicht ausgeführt. Die Windows werden jedoch ausgeführt.
- Enthält das Programm eine aktive Funktion "SUSPEND I/O", dann werden der Eingabezyklus im aktuellen Zyklus und der Ausgabezyklus im nächsten Zyklus nicht ausgeführt.

Tabelle B-1. Komponenten des CPU-Zyklus

Aufgabe	Beschreibung
Organisations-aufgaben	Eine Gruppe von Aufgaben (z.B. Aktualisierung der Zeitgeberwerte), die schnell zu Beginn jedes Zyklus durchgeführt werden.
Eingabezyklus	Während des Eingabezyklus fordert die CPU zuerst eine Eingangsaktualisierung von allen Genius-E/A-Buscontrollern an. Während sie auf die entsprechende Antwort wartet, liest die CPU die neuesten Eingaben von den E/A-Modulen im Chassis. Die Eingänge werden von Hauptchassis bis Chassis 7 und dort jeweils von der niedrigsten zur höchsten Steckplatznummer abgefragt. Während des Eingabezyklus werden keine Eingaben von VME-Modulen angenommen, diese erfolgen als Antwort auf die Anweisung VME_RD während der Programmausführung. Nachdem die CPU die Eingänge von den E/A-Modulen erfaßt hat, liest sie die Eingänge der Buscontroller. Die von den Buscontrollern empfangenen Eingangsdaten beinhalten auch andere Meldungen und Maschineneingaben. Die Reaktion auf diese Meldungen erfolgt während der Zeit, die dem System-Kommunikationswindow zugeteilt ist. Reagiert ein Buscontroller nicht rechtzeitig, oder stellt die CPU fest, daß ein Buscontroller die letzte Ausgangsaktualisierung noch nicht beendet hat, dann werden die Eingänge nicht gelesen und entsprechende Fehler in die E/A-Fehlertabelle eingetragen. Der Eingabezyklus wird nicht durchgeführt, wenn im Programm im vorhergehenden Zyklus eine SUSPEND I/O-Funktion aktiv war.
Programm-bearbeitung	Nach dem Eingabezyklus führt die CPU das Anwenderprogramm aus, wobei die neuesten Eingabedaten verwendet werden. Die Programmbearbeitung ergibt einen neuen Satz Ausgabedaten. Die Programmbearbeitung kann auf zahlreiche Arten so gesteuert werden, daß sie den Zeitbedingungen des Systems gerecht wird. Zum Beispiel: **1.** Die Programmlogik kann so strukturiert werden, daß Programmblöcke je nach Wichtigkeit oder Zeitbedingung mit unterschiedlicher Häufigkeit aufgerufen werden. **2.** Mit den Funktionen MCR oder JUMP können Programmteile übersprungen werden. **3.** Mit der Funktion SUSPEND I/O können für die Dauer eines Zyklus Ein- und Ausgabezyklus angehalten werden. Die E/A kann bei Bedarf über die Funktion DO I/O während der Programmausführung aktualisiert werden. **4.** Mit der Funktion SVCREQ können die den Windowanteilen des Zyklus zugeordneten Zeitscheiben aufgehoben oder verändert werden. **5.** Mit der Funktion SVCREQ kann die Zeitüberwachung (Watchdog) vorübergehend rückgesetzt werden. Tritt ein schwerwiegendes Problem auf, dann versetzt diese Zeitüberwachung die CPU in eine herabgesetzte Form des STOP-Modus, in der ein Fehler in der SPS-Fehlertabelle eingetragen wird und die Ausgänge auf ihre Vorgabezustände gesetzt werden. Die CPU kommuniziert dann nur noch mit dem Programmiergerät, sonstiger Datenverkehr ist nicht mehr möglich. Die Zeitüberwachung ist auf einen Standardwert von 200 ms eingestellt. Dieser Wert kann bei Bedarf in den Monitorfunktionen der SPS-Programmiersoftware über das Menü "SPS-Zyklussteuerung und -Uberwachung" verändert werden.

Tabelle B-1. Komponenten des CPU-Zyklus (Fortsetzung)

Aufgabe	Beschreibung
Ausgabezyklus	Während des Ausgabezyklus sendet die CPU Ausgabedaten zu den Genius-Buscontrollern. Die CPU wartet nicht auf eine Antwort von den Buscontrollern. Reagiert ein Buscontroller, dann wird die Antwort gespeichert und beim nächsten Eingabezyklus überprüft. Die zu den Buscontrollern gesendeten Ausgangsdaten beinhalten auch andere Meldungen und Maschinenausgaben. Diese Meldungen werden während der Zeitscheibe, die dem System-Kommunikationswindow zugeteilt ist, erzeugt. Nachdem die CPU die Ausgangsdaten zu den Buscontrollern gesendet hat, aktualisiert sie die Ausgänge der in den Chassis eingebauten E/A-Platinen. Hiervon sind die VME-Module nicht betroffen. Die Ausgänge dieser Module werden mit der Anweisung VME_WR im Programm aktualisiert. Die Diagnosedaten werden ebenfalls dann aktualisiert, wenn die einzelnen Buscontroller und E/A-Module bedient werden. Die Ausgänge werden auf die gleiche Art wie die Eingänge aktualisiert, vom Hauptchassis nach Chassis 7, jeweils mit der niedrigsten Steckplatznummer beginnend. Nachdem die CPU nach dem ersten Ausgabezyklus in die Betriebsart RUN/ENABLE umgeschaltet hat, befiehlt die CPU den einzelnen Buscontrollern und E/A-Modulen, die Ausgänge freizugeben, nachdem die neuen Ausgangswerte übertragen wurden. Der Ausgabezyklus wird nicht durchgeführt, wenn im Programm im vorhergehenden Zyklus eine SUSPEND I/O-Funktion aktiv war.
Abfrage fehlerhafter Module	Dieser Zyklusteil, der normalerweise übersprungen wird, wird durchlaufen, wenn die CPU beim Abfragen der Ausgänge ein fehlerhaftes Modul erkannt hat. Die CPU frägt in jedem Zyklus ein fehlerhaftes Modul ab. Ist das Modul noch fehlerhaft, dann erfolgt keine Reaktion. Wurde das Modul ersetzt, dann wird es im System neu konfiguriert. Wie nachstehend erläutert, findet die Neukonfiguration während des Programmiergeräte-Window statt.
Prüfsumme berechnen	Dieser Teil des CPU-Zyklus stellt sicher, daß die Prüfsummenberechnung des Programms, die normalerweise während eines früheren Zyklusteils im Hintergrund abläuft, beendet werden kann. Die gleiche Berechnung kann auch programmgesteuert während des Hintergrund-Windows ablaufen (siehe unten).
Planung des nächsten Zyklus	Hier legt die CPU fest, ob Programmiergeräte-Window, Systemkommunikations-Window oder Hintergrund-Window ablaufen sollen und wieviel Zeit den einzelenen Windows zugeordnet wird. Diese Entscheidung wird auf der Grundlage des gerade aktiven CPU-Zyklusmodus gefällt.
Programmiergeräte-Window	Ist ein Programmiergerät angeschlossen oder ist im System ein Modul, das neu konfiguriert werden muß (und bei der Abfrage fehlerhafter Module erkannt wurde), dann führt die CPU das Programmiergeräte-Window aus. Das Programmiergeräte-Window läuft nicht ab, wenn kein Programmiergerät angeschlossen ist und kein Modul im System neu konfiguriert werden muß. Während des Programmiergeräte-Window besitzt die Modul-Neukonfiguration höchste Priorität. Module werden innerhalb der dem Programmiergeräte-Window zugeteilten Zeit nach Bedarf neu konfiguriert. Gewöhnlich wird das Programmiergeräte-Window für den Datenaustausch zwischen CPU und Programmiergerät verwendet. Die CPU schließt zunächst unvollendete Anforderungen ab und bearbeitet dann die offenen Anfragen aus der Warteschlange. Zuletzt werden neue Anforderungen bearbeitet. Die Bearbeitung wird angehalten, wenn die dem Window zugeteilte Zeitscheibe abgelaufen ist. Die Zeit für das Programmiergeräte-Window ist standardmäßig auf 10 ms eingestellt. Dieser Wert kann bei Bedarf in der SPS-Programmiersoftware über das Menü "SPS-Zyklussteuerung und -Uberwachung" im Bereich zwischen 0 und 255 ms verändert werden. Bleibt nach Abschluß aller Aufgaben dieses Windows noch Zeit übrig, dann wird diese dem nächsten Window zugeteilt. Zeitdauer und Ausführung des Programmiergeräte-Windows können auch programmgesteuert über die Funktion SVCREQ #3 ablaufen. Hierdurch können die meisten Funktionen des Programmiergeräte-Windows während bestimmten zeitkritischen Zyklen übersprungen werden. Selbst wenn das Programmiergeräte-Window übersprungen wurde, kann die SPS noch auf Befehle zur Betriebsart- oder Zustandsänderung reagieren oder das Programmiergeräte-Window neu definieren.

Tabelle B-1. Komponenten des CPU-Zyklus (Fortsetzung)

Aufgabe	Beschreibung
System-Kommunikationswindow	Uber das System-Kommunikationswindow wird der Datenverkehr zwischen der CPU und intelligenten Modulen (z.B. PCM) abgewickelt. Der Datenverkehr mit den Genius-Buscontrollern wird während der Ein- und Ausgabezyklen abgewickelt. Meldungen an Genius-Buscontroller, die nicht die E/A betreffen, werden während des System-Kommunikationswindows verarbeitet. Zu Beginn des System-Kommunikationswindows beendet die CPU zunächst unvollendete Anforderungen und bearbeitet dann die offenen Anfragen aus der Warteschlange. Zuletzt werden neue Anforderungen bearbeitet. Die Bearbeitung wird angehalten, wenn die dem Window zugeteilte Zeitscheibe abgelaufen ist. Die Zeit für das System-Kommunikationswindow ist standardmäßig auf 10 ms eingestellt. Dieser Wert kann bei Bedarf in der SPS-Programmiersoftware über das Menü "SPS-Zyklussteuerung und -Uberwachung" im Bereich zwischen 0 und 255 ms verändert werden. Bleibt nach Abschluß aller Aufgaben noch Zeit für das Window übrig, dann wird diese dem nächsten Window zugeteilt. Zeitdauer und Ausführung des System-Kommunikationswindows können auch programmgesteuert über die Funktion SVCREQ #4 ablaufen. Hierdurch können Kommunikationsfunktionen während bestimmten zeitkritischen Zyklen übersprungen werden.

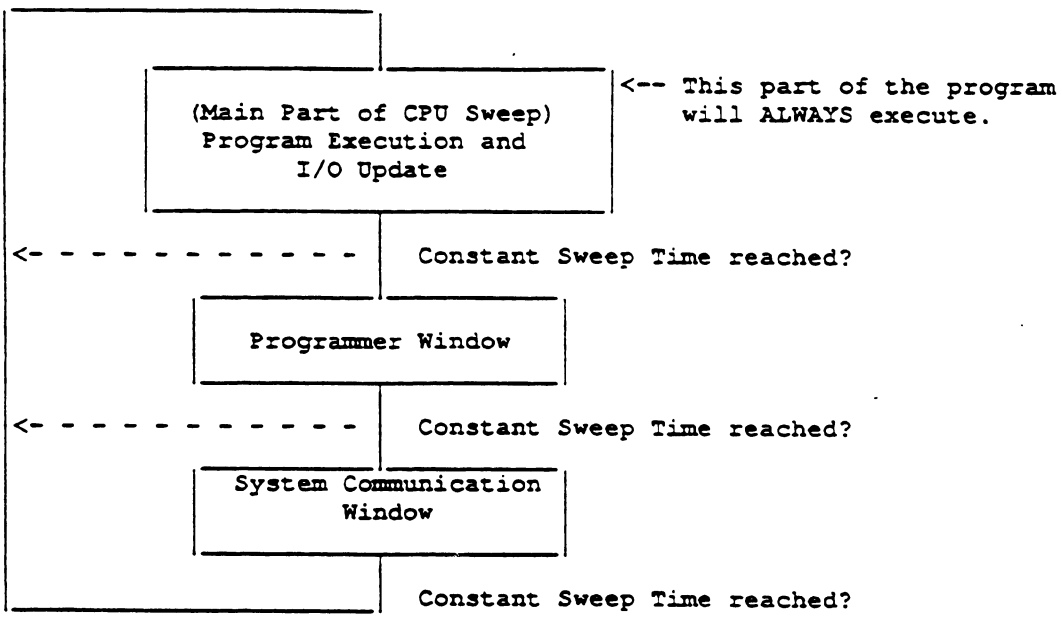
Konstante Zyklusdauer

Im Standard-Programmzyklus wird jeder Zyklus so schnell wie möglich abgearbeitet, wodurch sich für jeden einzelnen Zyklus eine unterschiedliche Zyklusdauer ergibt. Eine Alternative hierzu bildet die Betriebsart konstante Zyklusdauer. In diesem Modus belegt jeder Zyklus die gleiche Zyklusdauer, die im Menü "SPS-Zyklussteuerung und Überwachung" mit der Programmiersoftware auf Werte zwischen 5 und 200 Millisekunden eingestellt werden kann (hier muß ein Wert eingegeben werden, es gibt keine Voreinstellung).

Darüberhinaus kann mit der Funktion SVCREQ #1 die konstante Zyklusdauer programmgesteuert ein- und ausgeschaltet sowie der Zeitwert verändert werden. Diese Funktion zusammen mit der Fehlerreferenz OV_SWP ermöglicht eine programmgesteuerte Reaktion auf einen Zyklusüberlauf.

Einer der Gründe, eine konstante Zykluszeit zu wählen, liegt darin, daß die E/A in konstanten Zeitabständen aktualisiert werden soll. Ein anderer Grund kann darin liegen, daß genügend Zeit zwischen einem Ausgabezyklus und dem nächsten Eingabezyklus verstreichen muß, damit die Eingabegeräte sich nach dem Empfang der Ausgabedaten einschwingen können.

Wurde bei konstanter Zyklusdauer das Ende der Zykluszeit erreicht, dann können am Ende des Zyklus einige oder alle Windows übersprungen werden, wenn die eingestellte Zeit für eine Bearbeitung nicht ausgereicht hat. Programmbearbeitung und E/A-Aktualisierung werden jedoch immer normal ausgeführt.



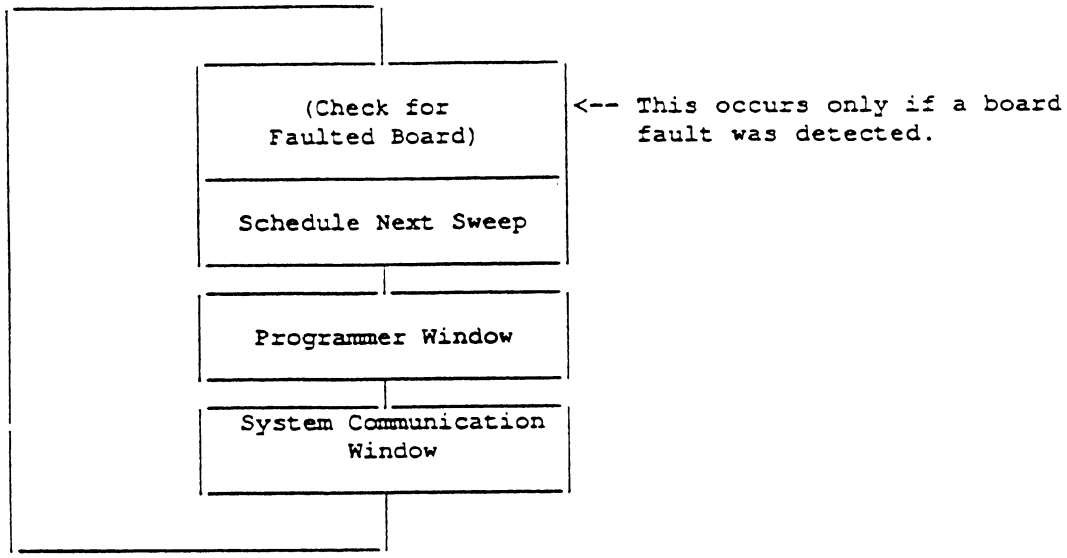
Läuft das Zeitglied während der Windowbearbeitung ab, dann wird das momentan bearbeitete Window beendet und der nächste CPU-Zyklus beginnt.

Wurde die eingestellte konstante Zykluszeit am Ende des letzten Windows noch nicht ausgeschöpft, dann werden die Windows wiederholt.

Dauern die Pflichtteile (Programmbearbeitung, E/A-Aktualisierung und weitere Aufgaben) *länger* als die konstante Zyklusdauer, dann werden sie beendet und es wird ein Zyklusüberlauffehler beim ersten Auftreten des Zyklusüberlaufs erzeugt. Hierauf wird die Fehlerreferenz OV_SWP gesetzt. Besteht ein Zyklusüberlauf, dann werden bestimmte Anforderungen vom Programmiergerät (z.B. Beenden der konstanten Zyklusdauer) trotzdem noch erkannt, obwohl die Zeitscheibe für das Programmiergeräte-Window bereits abgelaufen ist.

CPU-Zyklus in STOP-Modus

Ist die SPS im STOP-Modus, dann werden Programmbearbeitung und E/A-Aktualisierung nicht durchgeführt. Die Abfrage fehlerhafter Module wird fortgesetzt und Programmiergeräte-Window und System-Kommunikationswindow laufen ab. Enthält das System fehlerhafte Module, dann wird die Neukonfiguration wie üblich während des Programmiergeräte-Windows durchgeführt.



Ist der STOP-Modus das Ergebnis einer abgelaufenen Zeitüberwachung, dann führt die CPU Datenaustausch nur mit dem Programmiergerät durch; keine andere Kommunikation oder Operation ist möglich.

ANHANG C

Im Programmanzeige-/Editiermodus können Sie auf eine schnelle Art Programmanweisungen eingeben oder suchen, indem Sie ein "&"-Zeichen und die mnemonische Bezeichnung der Anweisung eingeben. Bei einigen Anweisungen können Sie auch eine Referenzadresse oder eine symbolische Adresse, einen Kennsatz (Label), oder eine Speicher-Referenzadresse eingeben.

Die Tabellen in diesem Anhang enthalten die mnemonischen Bezeichnungen sämtlicher Programmanweisungen von Logicmaster 90-70. Die komplette Mnemonik steht in Spalte 3 der Tabelle, die kürzestmögliche Eingabe in Spalte 4.

Während der Programmierung können Sie jederzeit ein Hilfsmenü mit diesen mnemonischen Bezeichnungen aufrufen, indem Sie die Tastenkombination ALT-I drücken.

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Kontakte	Kontakt: suchen nach beliebigem Kontakt	&CON	&CON
	Schließerkontakt -] [-	&NOCON	&NOCON
	Öffnerkontakt -]/[-	&NCCON	&NCCON
	Kontakt für pos. Übergang -]↑[-	&PCON	&PCON
	Kontakt für neg. Übergang -]↓[-	&NCON	&NCON
	Kontakt "Fehler" -[FAULT]-	&FAULT	&FA
	Kontakt "kein Fehler" -[NOFLT]-	&NOFAULT	&NOF
	Kontakt "oberer Grenzwert" -[HIALR]-	&HIALR	&HI
	Kontakt "unterer Grenzwert" -[LOALR]-	&LOALR	&LO
Spulen	Spulen: suchen nach beliebiger Spule	&COil	&COI
	Schließer -( )-	&NOCOil	&NOCOI
	Negiert -(/)-	&NCCOil	&NCCOi
	Positiver Übergang -(↑)-	&PCOil	&PCOi
	Negativer Übergang -(↓)-	&PCOil	&PCOi
	SET -(S)-	&SLat	&SL
	RESET -(R)-	&RLat	&RL
	Remanent -(M)-	&NOMcoil	&NOM
	Negiert remanent -(/M)-	&NCMcoil	&NCM
	SET, remanent -(SM)-	&SMlat	&SM
	RESET, remanent -(RM)-	&RMlat	&RM
Verbindungen	Horizontalverbindung ------	&HOrz	&HO
	Vertikalverbindung |	&VErt	&VE
Zeitglieder	Zeitglied zur Einschaltverzögerung	&ONdtr	&ON
	Zeitglied zur Ausschaltverzögerung	&OFdt	&OF
	Zeitglied zur Zeiterfassung	&TMr	&TM
Zähler	Aufwärtszähler	&UPctr	&UP
	Abwärtszähler	&DNctr	&DN

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Arithmetische Funktionen	Addition: beliebig	&ADD	&AD
	mit vorzeichenbehafteter ganzer Zahl	&ADD_INT	&AD_I
	mit vorzeichenloser ganzer Zahl	&ADD_UINT	&AD_U
	mit doppeltgenauer ganzer Zahl	&ADD_DINT	&AD_DI
	mit reeller Zahl	&ADD_REAL	&AD_R
	Subtraktion: beliebig	&SUB	&SUB
	mit vorzeichenbehafteter ganzer Zahl	&SUB_INT	&SUB_I
	mit vorzeichenloser ganzer Zahl	&SUB_UINT	&SUB_U
	mit doppeltgenauer ganzer Zahl	&SUB_DINT	&SUB_DI
	mit reeller Zahl	&SUB_REAL	&SUB_R
	Multiplikation: beliebig	&MUL	&MU
	mit vorzeichenbehafteter ganzer Zahl	&MUL_INT	&MU_I
	mit vorzeichenloser ganzer Zahl	&MUL_UINT	&MU_U
	mit doppeltgenauer ganzer Zahl	&MUL_DINT	&MU_DI
	mit reeller Zahl	&MUL_REAL	&MU_R
	mit gemischten ganzen Zahlen	&MUL_MIXED	&MU_M
	Division: beliebig	&DIV	&DIV
	mit vorzeichenbehafteter ganzer Zahl	&DIV_INT	&DIV_I
	mit vorzeichenloser ganzer Zahl	&DIV_UINT	&DIV_U
	mit doppeltgenauer ganzer Zahl	&DIV_DINT	&DIV_DI
	mit reeller Zahl	&DIV_REAL	&DIV_R
	mit gemischten ganzen Zahlen	&DIV_MIXED	&DIV_M
	Modulo: beliebig	&MOD	&MOD
	mit vorzeichenbehafteter ganzer Zahl	&MOD_INT	&MOD_I
	mit vorzeichenloser ganzer Zahl	&MOD_UINT	&MOD_U
	mit doppeltgenauer ganzer Zahl	&MOD_DINT	&MOD_DI
	Wurzel: beliebig	&SQRT	&SQRT
	mit vorzeichenbehafteter ganzer Zahl	&SQRT_INT	&SQRT
	mit doppeltgenauer ganzer Zahl	&SQRT_DINT	&SQRT_DI
	mit reeller Zahl	&SQRT_REAL	&SQRT_R
	Absolutwert: beliebig	&ABS	&ABS
	mit vorzeichenbehafteter ganzer Zahl	&ABS_INT	&ABS
	mit doppeltgenauer ganzer Zahl	&ABS_DINT	&ABS_DI
	mit reeller Zahl	&ABS_REAL	&ABS_R
	Trigonometrische Funktionen		
	Sinus	&SIN	&SIN
	Cosinus	&COS	&COS
	Tangens	&TAN	&TAN
	Arcussinus	&ASIN	&ASIN
	Arcuscosinus	&ACOS	&ACOS
	Arcustangens	&ATAN	&ATAN
	Logarithmische und Exponentialfunktionen		
	Dekadischer Logarithmus	&LOG	&LOG
	Natürlicher Logarithmus	&LN	&LN
	Potenz von *e*	&EXP	&EXP
	Potenz von x	&EXPT	&EXPT
	Bogenmaßkonvertierung		
	Konvertierung in Grad	&DEG	&DEG
	Konvertierung in Radiant	&RAD	&RAD

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Relationale Funktionen	Gleich: beliebig	&EQ	&EQ
	mit vorzeichenbehafteter ganzer Zahl	&EQ_INT	&EQ_I
	mit vorzeichenloser ganzer Zahl	&EQ_UINT	&EQ_U
	mit doppeltgenauer ganzer Zahl	&EQ_DINT	&EQ_DI
	mit reeller Zahl	&EQ_REAL	&EQ_R
	Ungleich: beliebig	&NE	&NE
	mit vorzeichenbehafteter ganzer Zahl	&NE_INT	&NE_I
	mit vorzeichenloser ganzer Zahl	&NE_UINT	&NE_U
	mit doppeltgenauer ganzer Zahl	&NE_DINT	&NE_DI
	mit reeller Zahl	&NE_REAL	&NE_R
	Größer als: beliebig	&GT	&GT
	mit vorzeichenbehafteter ganzer Zahl	&GT_INT	&GT_I
	mit vorzeichenloser ganzer Zahl	&GT_UINT	&GT_U
	mit doppeltgenauer ganzer Zahl	&GT_DINT	&GT_DI
	mit reeller Zahl	&GT_REAL	&GT_R
	Größer/gleich: beliebig	&GE	&GE
	mit vorzeichenbehafteter ganzer Zahl	&GE_INT	&GE_I
	mit vorzeichenloser ganzer Zahl	&GE_UINT	&GE_U
	mit doppeltgenauer ganzer Zahl	&GE_DINT	&GE_DI
	mit reeller Zahl	&GE_REAL	&GE_R
	Kleiner als: beliebig	&LT	&LT
	mit vorzeichenbehafteter ganzer Zahl	&LT_INT	&LT_I
	mit vorzeichenloser ganzer Zahl	&LT_UINT	&LT_U
	mit doppeltgenauer ganzer Zahl	&LT_DINT	&LT_DI
	mit reeller Zahl	&LT_REAL	&LT_R
	Kleiner/gleich: beliebig	&LE	&LE
	mit vorzeichenbehafteter ganzer Zahl	&LE_INT	&LE_I
	mit vorzeichenloser ganzer Zahl	&LE_UINT	&LE_U
	mit doppeltgenauer ganzer Zahl	&LE_DINT	&LE_DI
	mit reeller Zahl	&LE_REAL	&LE_R
	Vergleich: beliebig	&CMP	&CMP
	mit vorzeichenbehafteter ganzer Zahl	&CMP_INT	&CMP_I
	mit vorzeichenloser ganzer Zahl	&CMP_UINT	&CMP_U
	mit doppeltgenauer ganzer Zahl	&CMP_DINT	&CMP_DI
	mit reeller Zahl	&CMP_REAL	&CMP_R

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Bitoperationen	AND: beliebig	&AND	&AND
	Wort	&AND_WORD	&AND_W
	Doppelwort	&AND_DWORD	&AND_DW
	OR: beliebig	&OR	&OR
	Wort	&OR_WORD	&OR_W
	Doppelwort	&OR_DWORD	&OR_DW
	EXOR: beliebig	&XOR	&XOR
	Wort	&XOR_WORD	&XOR_W
	Doppelwort	&XOR_DWORD	&XOR_DW
	NOT: beliebig	&NOT	&NOT
	Wort	&NOT_WORD	&NOT_W
	Doppelwort	&NOT_DWORD	&NOT_DW
	Bit nach links verschieben: beliebig	&SHL	&SHL
	Wort	&SHL_W	&SHL_W
	Doppelwort	&SHL_DW	&SHL_DW
	Bit nach rechts verschieben: beliebig	&SHR	&SHR
	Wort	&SHr_W	&SHR_W
	Doppelwort	&SHr_DW	&SHR_DW
	Bit nach links rotieren: beliebig	&ROL	&ROL
	Wort	&rol_W	&rol_W
	Doppelwort	&rol_DW	&rol_DW
	Bit nach rechts rotieren: beliebig	&ROR	&ROR
	Wort	&ror_W	&ror_W
	Doppelwort	&ror_DW	&ror_DW
	Bit testen: beliebig	&BTST	&BT
	Wort	&BTST_W	&BT_W
	Doppelwort	&BTST_DW	&BT_DW
	Bit setzen: beliebig	&BSET	&BS
	Wort	&BSET_W	&BS_W
	Doppelwort	&BSET_DW	&BS_DW
	Bit löschen: beliebig	&BCLR	&BC
	Wort	&BCLR_W	&BC_W
	Doppelwort	&BCLR_DW	&BC_DW
	Bitposition: beliebig	&BPOS	&BPOS
	Wort	&BPOS_W	&BPOS_W
	Doppelwort	&BPOS_DW	&BPOS_DW
	Maskierter Vergleich: beliebig	&MCMP	&MCM
	Wort	&MCMP_W	&MCM_W
	Doppelwort	&MCMP_DW	&MCM_DW

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Daten-verschiebungs-funktionen	Kopieren: beliebig	&MOVE	&MOV
	mit vorzeichenbehafteter ganzer Zahl	&MOVE_INT	&MOV_I
	mit vorzeichenloser ganzer Zahl	&MOVE_UINT	&MOV_UI
	mit doppeltgenauer ganzer Zahl	&MOVE_DINT	&MOV_DI
	bitweise	&MOVE_BIT	&MOV_BI
	mit Wort	&MOVE_W	&MOV_W
	mit Doppelwort	&MOVE_DW	&MOV_DW
	mit reeller Zahl	&MOVE_REAL	&MOV_R
	Block verschieben: beliebig	&BLKMOV	&BLKM
	mit vorzeichenbehafteter ganzer Zahl	&BLKMOV_INT	&BLKM_I
	mit vorzeichenloser ganzer Zahl	&BLKMOV_UINT	&BLKM_UI
	mit doppeltgenauer ganzer Zahl	&BLKMOV_DINT	&BLKM_DI
	mit Wort	&BLKMOV_W	&BLKM_W
	mit Doppelwort	&BLKMOV_DW	&BLKM_DW
	mit reeller Zahl	&BLKMOV_REAL	&BLKM_R
	Block löschen: Wort	&BLKCLR	&BLKCL
	Schieberegister: beliebig	&SHFR	&SHFR
	Bit	&SHFR_BIT	&SHFR_BI
	Wort	&SHFR_WORD	&SHFR_W
	Doppelwort	&SHFR_DWORD	&SHFR_DW
	Bitfolgesteuerung: beliebig	&BITSEQ	&BITSEQ
	Kommunikationsanforderung	&COMMREQ	&COMMR
	VME lesen: beliebig	&VMERD	&VMERD
	Byte lesen	&VMERD_BY	&VMERD_BY
	Wort lesen	&VMERD_W	&VMERD_W
	VME schreiben: beliebig	&VMEWRT	&VMEW
	Byte schreiben	&VMEWRT_BY	&VMEW_BY
	Wort schreiben	&VMEWRT_W	&VMEW_W
	VME lesen/ändern/schreiben: beliebig	&VMERMW	&VMERM
	Byte lesen/ändern/schreiben	&VMERMW_BY	&VMERM_BY
	Wort lesen/ändern/schreiben	&VMERMW_W	&VMERM_W
	VME testen: beliebig	&VMETST	&VMET
	Byte testen	&VMETST_BY	&VMET_BY
	Wort testen	&VMETST_W	&VMET_W
	Byte vertauschen: beliebig	&SWAP	&SW
	Byte vertauschen: Wort	&SWAP_W	&SW_W
	Byte vertauschen: Doppelwort	&SWAP_DW	&SW_DW

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Konvertierungs-funktionen	Konvertierung in vorzeichenbeh. ganze Zahl: beliebig	&INT	&I
	vorzeichenlose in vorzeichenbeh. ganze Zahl	&INT	&I
	doppeltgenaue in vorzeichenbeh. ganze Zahl	&INT_DINT	&I_DI
	BCD-4 in vorzeichenbehaftete ganze Zahl	&INT_BCD4	&I_BCD4
	reelle in vorzeichenbehaftete ganze Zahl	&INT_REAL	&I_R
	Konvertierung in vorzeichenlose ganze Zahl: beliebig	&UNIT	&UI
	vorzeichenbeh. in vorzeichenlose ganze Zahl	&UNIT	&UI
	doppeltgenaue in vorzeichenlose ganze Zahl	&UNIT_DINT	&_UI
	BCD-4 in vorzeichenlose ganze Zahl	&UNIT_BCD4	&_BCD8
	reelle in vorzeichenlose ganze Zahl	&UINT_REAL	&UI_R
	Konvertierung in doppeltgenaue ganze Zahl: beliebig	&DINT	&DIN
	vorzeichenbeh. in doppeltgenaue ganze Zahl	&DINT	&DIN
	vorzeichenlose in doppeltgenaue ganze Zahl	&DINT_UINT	&DIN_UI
	BCD-8 in doppeltgenaue ganze Zahl	&DINT_BCD8	&DIN_BCD8
	reelle in doppeltgenaue ganze Zahl	&DINT_REAL	%DIN_R
	Konvertierung in BCD-4: beliebig	&BCD4	&BCD4
	vorzeichenbehaftete ganze Zahl in BCD-4	&BCD4	&BCD4
	vorzeichenlose ganze Zahl in BCD-4	&BCD4_UINT	&_UI
	Konvertierung doppeltgenaue ganze Zahl in BCD-8	&BCD8	&BCD8
	Konvertierung in reelle Zahl: beliebig	&REAL	&R
	vorzeichenbehaftete in reelle Zahl	&REAL	&R
	vorzeichenlose in reelle Zahl	&REAL_UINT	&R_UI
	doppeltgenaue in reelle Zahl	&REAL_DINT	&R_DI
	BCD-4 in reelle Zahl	&REAL_BCD4	&R_BCD4
	BCD-8 in reelle Zahl	&REAL_BCD8	&R_BCD8
	Runden: beliebig	&TRINT	&TRINT
	Runden auf vorzeichenbehaftete ganze Zahl	&TRINT	&TRINT
	Runden auf vorzeichenlose ganze Zahl	&TRDINT	&TRDINT
Steuerfunktionen	Programmblock aufrufen	&CALL	&CA
	DO I/O	&DOio	&DO
	E/A-Aktualisierung unterdrücken	&SUSIO	&SUS
	Hauptsteuerrelais	&MCR	&MCR
	Hauptsteuerrelais-Ende	&ENdmcr	&EN
	JUMP	&Jump	&J
	LABEL	&LAbel	&LA
	Strompfad-Kommentar	&COMMEnt	&COMME
	System-Bedienaufruf	&SVCREQ	&SVC
	PID-ISA-Algorithmus: beliebig	&PIDISA	&PIDISA
	PID-IND-Algorithmus: beliebig	&PIDIND	&PIDIND

Funktion	Anweisung	Vollständige Mnemonik	Gekürzte Mnemonik
Tabellen-funktionen	Tabelle lesen: beliebig	&TBLRD	&TBLR
	Tabelle lesen, Wort	&TBLRD_W	&TBLR_W
	Tabelle lesen, Doppelwort	&TBLRD_DW	&TBLR_DW
	Tabelle lesen, ganze Zahl mit Vz.	&TBLRD_INT	&TBLR_I
	Tabelle lesen, ganze Zahl ohne Vz.	&TBLRD_UINT	&TBLR_UI
	Tabelle lesen, doppeltgenaue ganze Zahl	&TBLRD_DINT	&TBLR_DI
	Tabelle schreiben: beliebig	&TBLWRT	&TBLW
	Tabelle schreiben, Wort	&TBLWRT_W	&TBLW_W
	Tabelle schreiben, Doppelwort	&TBLWRT_DW	&TBLW_DW
	Tabelle schreiben, ganze Zahl mit Vz.	&TBLWRT_INT	&TBLW_I
	Tabelle schreiben, ganze Zahl ohne Vz.	&TBLWRT_UINT	&TBLW_UI
	Tabelle schreiben, doppeltgenaue ganze Zahl	&TBLWRT_DINT	&TBLW_DI
	LIFO lesen: beliebig	&LIFORD	&LIFOR
	LIFO lesen, Wort	&LIFORD_W	&LIFOR_W
	LIFO lesen, Doppelwort	&LIFORD_DW	&LIFOR_DW
	LIFO lesen, ganze Zahl mit Vz.	&LIFORD_INT	&LIFOR_I
	LIFO lesen, ganze Zahl ohne Vz.	&LIFORD_UINT	&LIFOR_UI
	LIFO lesen, doppeltgenaue ganze Zahl	&LIFORD_DINT	&LIFOR_DI
	LIFO schreiben: beliebig	&LIFOWRT	&LIFOW
	LIFO schreiben, Wort	&LIFOWRT_W	&LIFOW_W
	LIFO schreiben, Doppelwort	&LIFOWRT_DW	&LIFOW_DW
	LIFO schreiben, ganze Zahl mit Vz.	&LIFOWRT_INT	&LIFOW_I
	LIFO schreiben, ganze Zahl ohne Vz.	&LIFOWRT_UINT	&LIFOW_UI
	LIFO schreiben, doppeltgenaue ganze Zahl	&LIFOWRT_DINT	&LIFOW_DI
	FIFO lesen: beliebig	&FIFORD	&FIFOR
	FIFO lesen, Wort	&FIFORD_W	&FIFOR_W
	FIFO lesen, Doppelwort	&FIFORD_DW	&FIFOR_DW
	FIFO lesen, ganze Zahl mit Vz.	&FIFORD_INT	&FIFOR_I
	FIFO lesen, ganze Zahl ohne Vz.	&FIFORD_UINT	&FIFOR_UI
	FIFO lesen, doppeltgenaue ganze Zahl	&FIFORD_DINT	&FIFOR_DI
	FIFO schreiben: beliebig	&FIFOWRT	&FIFOW
	FIFO schreiben, Wort	&FIFOWRT_W	&FIFOW_W
	FIFO schreiben, Doppelwort	&FIFOWRT_DW	&FIFOW_DW
	FIFO schreiben, ganze Zahl mit Vz.	&FIFOWRT_INT	&FIFOW_I
	FIFO schreiben, ganze Zahl ohne Vz.	&FIFOWRT_UINT	&FIFOW_UI
	FIFO schreiben, doppeltgenaue ganze Zahl	&FIFOWRT_DINT	&FIFOW_DI
	Sortieren: beliebig	&SORT	&SORT
	Sortieren, ganze Zahl mit Vz.	&SORT_INT	&SORT_I
	Sortieren, ganze Zahl ohne Vz.	&SORT_UINT	&SORT_UI
	Sortieren, Wort	&SORT_WORD	&SORT_W

ANHANG D

Tabelle D-1. Speicherbelegungs-Arbeitsblatt für die SPS Serie 90-70

Verwenden Sie dieses Arbeitsblatt, um für die einzelnen Typen die Gesamtmenge der verwendeten Speicherbytes (%R, %AI, %AQ, %P, %L) und zugehörigen Punktfehler (falls vorhanden) zu ermitteln. Addieren Sie dann die einzelnen Werte. Den für das Anwenderprogramm verfügbare Wert finden Sie, indem Sie die Gesamtsumme von dem in Ihrem System verwendeten Speicher abziehen.

%R belegt = ... verwendete Anzahl %R x 2 (Bytes) = + _____
(die Anzahl verwendeter %R ist je nach Konfiguration in Schritten von 1 k)

Hinweis: Wird %P verwendet, dann ist die Minimalbelegung 160.

Gesamt

%P belegt =

((Anzahl verwendeter %P/32 aufgerundet auf ganze Zahl)x32)+32 = _____ x 2 (Bytes) = _____

Beispiele:

%P verwendet	%P belegt (Worte)
1 bis 128	160
129 bis 160	192
161 bis 192	224
193 bis 224	256

Hinweis: Die folgenden Werte für %L gelten pro Programmblock.
Hinweis: Wird %L verwendet, dann ist die Minimalbelegung 96.

%L belegt =

((Anzahl verwendeter %L/32 aufgerundet auf ganze Zahl)x32)+32 = _____ x 2 (Bytes) = + _____

Beispiele:

%L verwendet	%L belegt (Worte)
1 bis 64	96
65 bis 96	128
97 bis 128	160
129 bis 192	192

Jeder %L- oder %P-Datenblock belegt zusätzlichen Platz (Overhead) im SPS-Speicher (ca. 31 Bytes plus 48 Bytes reserviert für Verwendung durch SPS).

Tabelle D-1. Speicherbelegungs-Arbeitsblatt für die SPS Serie 90-70 (Fortsetzung)

	+	
%AI belegt (Punktfehler gesperrt) = Anzahl %AI verwendet x 2 (Bytes)	=	_____
ODER		
	+	
%AI belegt (Punktfehler freigegeben) = Anzahl %AI verwendet x 3 (Bytes)	=	_____
%AQ belegt (Punktfehler gesperrt) = Anzahl %AQ verwendet x 2 (Bytes)	=	_____
ODER		
	+	
%AQ belegt (Punktfehler freigegeben) = Anzahl %AQ verwendet x 3 (Bytes)	=	_____
Sind Punktfehler freigegeben, addieren Sie		
128 Bytes für CPU 731/732		
oder 512 Bytes für CPU 771/772	+	
oder 3072 Bytes für CPU 781/782		_____
Gesamtsumme (Bytes)		_____
CPU-Speichergröße gesamt (Bytes):		
32 k = 32.768		
oder 64 k = 65.536		
oder 128 k = 131.072		
oder 256 k = 262.144		
oder 512 k = 524.288	=	
Subtrahieren Sie die Gesamtsumme (von oben)	-	_____
Verfügbarer Programmspeicher (in Bytes)	=	_____

ANHANG E

In diesem Anhang werden die in der Softwareumgebung aktiven Tastaturfunktionen beschrieben. Diese Informationen können Sie auch mit der Tastenkombination ALT-K auf dem Bildschirm anzeigen.

Diese Tasten sind im gesamten Softwarepaket verfügbar

Tasten	Funktion
ALT-A	Abbruch
ALT-C	Feld löschen
ALT-M	Programmiergeräte-Modus umschalten
ALT-R	SPS-Betriebsart (RUN/STOP) umschalten
ALT-E	Statusbereich umschalten
ALT-L	Plattendirectory anzeigen
ALT-P	Aktuellen Bildschirminhalt drucken
ALT-H	Kontextspezifische Hilfstexte anzeigen
ALT-K	Liste der Tastenfunktionen anzeigen
ALT-I	Hilfsanzeige, mnemonische Programmanweisungen
ALT-T	Tastatur-Lernmodus, Beginn
ALT-Q	Tastatur-Lernmodus, Ende
ALT-n	Playback-Datei n (n = 1...9)
F12	Diskrete Referenz umschalten
F11	Diskrete Referenz überschreiben
CTRL-Break	Logicmaster 90 verlassen, Rückkehr zu DOS
Esc	Zu höherer Funktionsebene umschalten
CTRL-Home	Vorherige Befehlszeile anzeigen
CTRL-End	Nächste Befehlszeile anzeigen
CTRL-←	Cursor nach links im Feld bewegen
CTRL-→	Cursor nach rechts im Feld bewegen
CTRL-D	Referenzadresse dekrementieren
CTRL-U	Referenzadresse inkrementieren
Tab	Vorwärts durch Eingabefelder springen
Shift-Tab	Rückwärts durch Eingabefelder springen
Enter	Feldinhalt übernehmen
CTRL-E	Anzeige des letzten Systemfehlers
Zehnertastatur -	Diskrete Referenz umschalten
Zehnertastatur +	Diskrete Referenz überschreiben

Diese Tasten sind nur im Programmeditor verfügbar

Tasten	Funktion
ALT-B	Glocke beim Texteditieren ein-/ausschalten
ALT-D	Programmanweisung unter Cursor löschen
ALT-S	Programm/Anderungen in SPS und auf Platte speichern
ALT-X	Anzeige Programmgröße
ALT-N	Editor-Anzeigemodus umschalten
ALT-U	Platteninhalt aktualisieren
ALT-V	Variablentabelle-Window
ALT-F2	Gehe zu Operanden-Referenztabelle
Zehnertastatur +	Strompfad übernehmen
Enter	Strompfad übernehmen
CTRL-PgUp	Vorherigen Strompfad anzeigen
CTRL-PgDn	Nächsten Strompfad anzeigen
~	Horizontalverbindung
|	Vertikalverbindung
Tab	Zum nächsten Operandenfeld springen
@Referenzadresse	Indirekte Referenzen

Spezialtasten

ALT-U	Platteninhalt aktualisieren. Nur in E/A- und CPU-Konfiguration verfügbar.
ALT-O	Paßwort überschreiben. Nur im Paßwortmenü der Konfigurationssoftware verfügbar.

INDEX

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Z