SIEMENS

SIMATIC

Kontaktplan (KOP) für S7-300/400

Referenzhandbuch

Dieses Referenzhandbuch ist Bestandteil des Dokumentationspaketes mit der Bestellnummer:

6ES7810-4CA08-8AW1

Ausgabe 03/2006
A5E00706948-01

Sicherheitshinweise

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

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

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

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

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

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

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

Qualifiziertes Personal

Das zugehörige Gerät/System darf nur in Verbindung mit dieser Dokumentation eingerichtet und betrieben werden. Inbetriebsetzung und Betrieb eines Gerätes/Systems dürfen nur von **qualifiziertem Personal** vorgenommen werden. Qualifiziertes Personal im Sinne der sicherheitstechnischen Hinweise dieser Dokumentation sind Personen, die die Berechtigung haben, Geräte, Systeme und Stromkreise gemäß den Standards der Sicherheitstechnik in Betrieb zu nehmen, zu erden und zu kennzeichnen.

Bestimmungsgemäßer Gebrauch

Beachten Sie Folgendes:

Warnung
Das Gerät darf nur für die im Katalog und in der technischen Beschreibung vorgesehenen Einsatzfälle und nur in Verbindung mit von Siemens empfohlenen bzw. zugelassenen Fremdgeräten und -komponenten verwendet werden. Der einwandfreie und sichere Betrieb des Produktes setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung und Montage sowie sorgfältige Bedienung und Instandhaltung voraus.

Marken

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

Haftungsausschluss

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

Siemens AG
Automation and Drives
Postfach 4848
90437 NÜRNBERG
DEUTSCHLAND

A5E00706948-01
03/2006

Vorwort

Zweck des Handbuchs

Dieses Handbuch unterstützt Sie bei der Erstellung von Anwenderprogrammen in der Programmiersprache KOP.

Es beschreibt die Sprachelemente der Programmiersprache KOP, ihre Syntax und Funktionsweise.

Erforderliche Grundkenntnisse

Dieses Handbuch richtet sich an Programmierer von S7-Programmen, Inbetriebsetzer und Servicepersonal.

Zum Verständnis des Handbuchs sind allgemeine Kenntnisse auf dem Gebiet der Automatisierungstechnik erforderlich.

Außerdem werden Kenntnisse über die Verwendung von Computern oder PC-ähnlichen Arbeitsmitteln (z. B. Programmiergeräten) unter den Betriebssystemen MS Windows 2000 Professional, MS Windows XP Professional oder MS Windows Server 2003 vorausgesetzt.

Gültigkeitsbereich des Handbuchs

Das Handbuch ist gültig für die Programmiersoftware STEP 7 ab Version 5.4.

Normerfüllung nach IEC 1131-3

KOP entspricht der in der Norm DIN EN-61131-3 (int. IEC 1131-3) festgelegten Sprache ”Kontaktplan” (engl. Ladder Diagram). Genaue Aussagen zur Normerfüllung finden Sie in der Normerfüllungstabelle in der NORM.TAB-Datei von STEP 7.

Dokumentationspakete zu STEP 7

Das vorliegende Handbuch zu KOP setzt theoretische Kenntnisse über S7-Programme voraus, die Sie in der Online-Hilfe zu STEP 7 nachlesen können. Da die Sprachpakete auf der Basissoftware STEP 7 aufsetzen, sollten Sie bereits Kenntnisse im Umgang mit der Basissoftware STEP 7 und deren Dokumentation haben.

Dieses Handbuch ist Bestandteil des Dokumentationspaketes "STEP 7 Referenzwissen“.

Die folgende Tabelle zeigt die Dokumentation zu STEP 7 im Überblick:

Handbücher	**Zweck**	**Bestellnummer**
STEP 7-Grundwissen mit • Erste Schritte und Übungen mit STEP 7 • Programmieren mit STEP 7 • Hardware konfigurieren und Verbindungen projektieren mit STEP 7 • Von S5 nach S7, Umsteigerhandbuch	Das Grundwissen für technisches Personal, das das Vorgehen zur Realisierung von Steuerungsaufgaben mit STEP 7 und S7-300/400 beschreibt.	6ES7810-4CA08-8AW0
STEP 7-Referenzwissen mit • Handbücher KOP/FUP/AWL für S7-300/400 • Standard- und Systemfunktionen für S7-300/400 Band 1 und Band 2	Das Referenzwissen zum Nachschlagen, das die Programmiersprachen KOP, FUP und AWL sowie Standard- und Systemfunktionen ergänzend zum STEP 7-Grundwissen beschreibt.	6ES7810-4CA08-8AW1

Online-Hilfen	**Zweck**	**Bestellnummer**
Hilfe zu STEP 7	Das Grundwissen zum Programmieren und Hardware konfigurieren mit STEP 7 als Online-Hilfe	Bestandteil des Softwarepaketes STEP 7
Referenzhilfen zu AWL/KOP/FUP Referenzhilfe zu SFBs/SFCs Referenzhilfe zu Organisationsbausteinen	Kontextsensitives Referenzwissen	Bestandteil des Softwarepaketes STEP 7

Online-Hilfe

Ergänzend zum Handbuch erhalten Sie bei der Nutzung der Software detaillierte Unterstützung durch die in die Software integrierte Online-Hilfe.

Auf die Inhalte der Online-Hilfe können Sie wie folgt zugreifen:

- Kontext-sensitive Hilfe zum markierten Objekt über Menübefehl **Hilfe > Hilfe zum Kontext** über Funktionstaste F1 oder über Fragezeichen in der Funktionsleiste.
- Hilfe zu STEP 7 über den Menübefehl **Hilfe > Hilfethemen** oder die Schaltfläche "Hilfe zu STEP 7" im Hilfefenster der kontext-sensitiven Hilfe.
- Glossar für alle STEP 7-Applikationen über die Schaltfläche "**Glossar"**.

Wenn Sie Informationen der Online-Hilfe lieber in gedruckter Form lesen möchten, können Sie einzelne Hilfethemen, Bücher oder die gesamte Hilfe auch ausdrucken.

Dieses Handbuch ist ein Auszug der "Hilfe zu KOP". Aufgrund der identischen Gliederungsstruktur von Handbuch und Online-Hilfe können Sie bequem zwischen Handbuch und Online-Hilfe wechseln.

Weitere Unterstützung

Bei Fragen zur Nutzung der im Handbuch beschriebenen Produkte, die Sie hier nicht beantwortet finden, wenden Sie sich bitte an Ihren Siemens-Ansprechpartner in den für Sie zuständigen Vertretungen und Geschäftsstellen.

Ihren Ansprechpartner finden Sie unter:

http://www.siemens.com/automation/partner

Den Wegweiser zum Angebot an technischen Dokumentationen für die einzelnen SIMATIC Produkte und Systeme finden Sie unter:

http://www.siemens.de/simatic-tech-doku-portal

Den Online-Katalog und das Online-Bestellsystem finden Sie unter:

http://mall.automation.siemens.com/

Trainingscenter

Um Ihnen den Einstieg in das Automatisierungssystem SIMATIC S7 zu erleichtern, bieten wir entsprechende Kurse an. Wenden Sie sich bitte an Ihr regionales Trainingscenter oder an das zentrale Trainingscenter in D 90327 Nürnberg.

Telefon: +49 (911) 895-3200.

Internet: http://www.sitrain.com

Technical Support

Sie erreichen den Technical Support für alle A&D-Produkte

- Über das Web-Formular für den Support Request
  http://www.siemens.de/automation/support-request
- Telefon: + 49 180 5050 222
- Fax: + 49 180 5050 223

Weitere Informationen zu unserem Technical Support finden Sie im Internet unter
http://www.siemens.de/automation/service

Service & Support im Internet

Zusätzlich zu unserem Dokumentations-Angebot bieten wir Ihnen im Internet unser komplettes Wissen online an.

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

Dort finden Sie:

- den Newsletter, der Sie ständig mit den aktuellsten Informationen zu Ihren Produkten versorgt.
- die für Sie richtigen Dokumente über unsere Suche in Service & Support.
- ein Forum, in welchem Anwender und Spezialisten weltweit Erfahrungen austauschen.
- Ihren Ansprechpartner für Automation & Drives vor Ort.
- Informationen über Vor-Ort Service, Reparaturen, Ersatzteile. Vieles mehr steht für Sie unter dem Begriff "Leistungen“ bereit

Inhaltsverzeichnis

1 Bitverknüpfung

1.1 Bitverknüpfungsoperationen Übersicht

Beschreibung

Bitverknüpfungsoperationen arbeiten mit den Zahlen "1" und "0". Diese Zahlen bilden die Basis des Dualsystems und werden "Binärziffern" oder kurz "Bits" genannt. Im Zusammenhang mit UND, ODER, XOR und Ausgängen steht eine "1" für "logisch JA" und eine "0" für "logisch NEIN".

Die Bitverknüpfungsoperationen interpretieren die Signalzustände "1" und "0" und verknüpfen sie entsprechend der Booleschen Logik. Die Verknüpfungen liefern ein Ergebnis von "1" oder "0", das sogenannte Verknüpfungsergebnis (VKE).

Folgende Bitverknüpfungsoperationen stehen Ihnen zur Verfügung:

- ---| |--- Schließerkontakt
- ---| / |--- Öffnerkontakt
- ---(SAVE) Verknüpfungsergebnis in BIE-Register laden
- XOR Exklusiv-ODER verknüpfen
- ---( ) Relaisspule, Ausgang
- ---(#)--- Konnektor
- ---|NOT|--- Verknüpfungsergebnis invertieren

Folgende Operationen reagieren auf ein VKE von "1":

- ---(S) Ausgang setzen
- ---(R) Ausgang rücksetzen
- SR Flipflop setzen rücksetzen
- RS Flipflop rücksetzen setzen

Einige Operationen reagieren auf einen steigenden oder fallenden Flankenwechsel, so daß Sie damit eine der folgenden Funktionen ausführen können:

- ---(N)--- Flanke 1 -> 0 abfragen
- ---(P)--- Flanke 0 -> 1 abfragen
- NEG Signalflanke 1 -> 0 abfragen
- POS Signalflanke 0 -> 1 abfragen
- Peripherie direkt lesen
- Peripherie direkt schreiben

1.2 ---| |--- Schließerkontakt

Symbol

<Operand>

---| |---

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D, T, Z	Abgefragtes Bit

Beschreibung

---| |--- (Schließerkontakt) wird geschlossen, wenn der Wert des abgefragten Bits, der am angegebenen **<Operanden>** gespeichert wird, gleich "1" ist. Wenn der Kontakt geschlossen ist, fließt der Strom über den Kontakt und das Verknüpfungsergebnis (VKE) ist "1" .

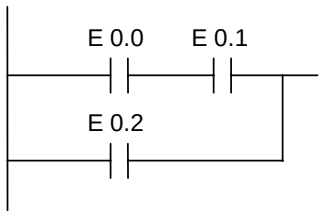
Andernfalls, wenn der Signalzustand am angegebenen **<Operanden>** "0" ist, ist der Kontakt geöffnet. Ist der Kontakt geöffnet, dann fließt kein Strom und das Verknüpfungsergebnis der Operation (VKE) ist "0".

Bei Reihenschaltungen wird der Kontakt **---| |---** mit dem VKE bitweise durch UND verknüpft. Bei Parallelschaltungen wird der Kontakt mit dem VKE durch ODER verknüpft.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



Strom kann fließen, wenn:
der Zustand an den Eingängen E 0.0 UND E 0.1 "1" ist ODER der Zustand an Eingang E 0.2 "1" ist.

1.3 ---| / |--- Öffnerkontakt

Symbol

<Operand>

—| / |—

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D, T, Z	Abgefragtes Bit

Beschreibung

---| / |--- (Öffnerkontakt) ist geschlossen, wenn der Wert des abgefragten Bits, der am angegebenen **<Operanden>** gespeichert wird, gleich "0" ist. Wenn der Kontakt geschlossen ist, fließt der Strom über den Kontakt und das Verknüpfungsergebnis (VKE) ist "1".

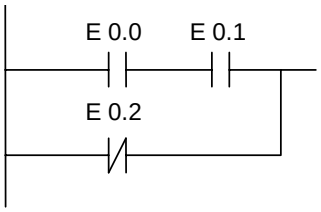
Andernfalls, wenn der Signalzustand am angegebenen **<Operanden>** "1" ist, ist der Kontakt geöffnet. Wenn der Kontakt geöffnet ist, dann fließt kein Strom und das Verknüpfungsergebnis der Operation (VKE) ist "0".

Bei Reihenschaltungen wird der Kontakt **---| / |---** mit dem VKE bitweise durch UND verknüpft. Bei Parallelschaltungen wird der Kontakt mit dem VKE durch ODER verknüpft.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel

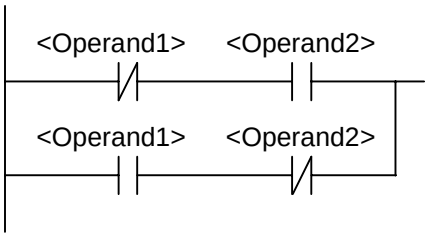


Strom kann fließen, wenn:
der Zustand an den Eingängen E 0.0 UND E 0.1 "1" ist ODER der Zustand an Eingang E 0.2 "0" ist.

1.4 XOR Exklusiv-ODER verknüpfen

Symbol

Für die Funktion **XOR** ist es erforderlich, ein Netzwerk von Öffnern und Schließern zu erstellen:

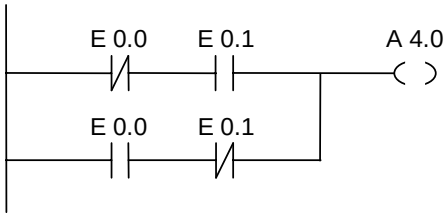


Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand1>	BOOL	E, A, M, L, D, T, Z	Abgefragtes Bit
<Operand2>	BOOL	E, A, M, L, D, T, Z	Abgefragtes Bit

Beschreibung

XOR (Exklusiv-ODER verknüpfen) erstellt ein VKE von "1", wenn der Signalzustand der beiden angegebenen Bits unterschiedlich ist.

Beispiel



Ausgang A 4.0 ist "1", wenn (E 0.0 = 0 UND E 0.1 = 1) ODER (E 0.0 = 1 UND E 0.1 = 0).

1.5 --|NOT|-- Verknüpfungergebnis invertieren

Symbol

---|NOT|---

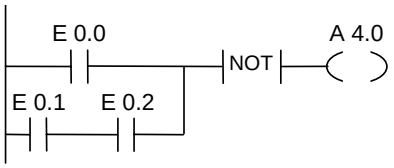
Beschreibung

---|NOT|--- (Verknüpfungsergebnis invertieren) invertiert das VKE-Bit.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	-	1	X	-

Beispiel



Ausgang A 4.0 ist "0", wenn:

der Zustand an Eingang E 0.0 "1" ODER der Zustand an E 0.1. UND E 0.2 "1" ist.

1.6 ---( ) Relaisspule, Ausgang

Symbol

<Operand>

---( )

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Zugeordnetes Bit

Beschreibung

---( ) (Relaisspule, Ausgang) funktioniert wie eine Spule in einem Stromlaufplan. Fließt Strom zur Spule (VKE = 1), wird das Bit am **<Operanden>** auf "1" gesetzt. Fließt kein Strom zur Spule (VKE = 0), wird das Bit am **<Operanden>** auf "0" gesetzt. Eine Ausgangsspule kann nur am rechten Ende eines Strompfads in einem Kontaktplan angeordnet werden. Mehrfachausgänge sind möglich (max. 16, siehe Beispiel). Ein negierter Ausgang kann mit der Operation ---|NOT|--- (Verknüpfungsergebnis invertieren) erstellt werden.

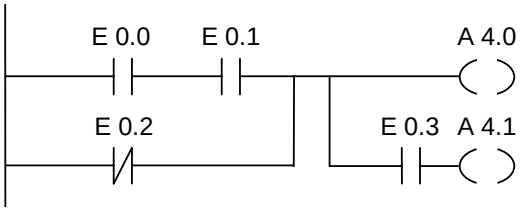
Abhängigkeit vom MCR (Master Control Relay)

Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich eine Ausgangsspule in einem aktiven MCR-Bereich befindet. Ist das MCR eingeschaltet und es fließt Strom zu einer Ausgangsspule, dann wird das adressierte Bit auf den aktuellen Signalzustand des Signalflusses gesetzt. Ist das MCR ausgeschaltet, wird eine "0" an den angegebenen Operanden geschrieben, unabhängig vom Signalzustand des Signalflusses.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	-	0

Beispiel



Ausgang A 4.0 ist "1", wenn:

(der Zustand an Eingang E 0.0 UND E 0.1 "1" ist) ODER der Zustand an Eingang E 0.2 "0" ist.

Ausgang A 4.1 ist "1", wenn:

(der Zustand an Eingang E 0.0 UND E 0.1 "1" ist ODER der Zustand an Eingang E 0.2 "0" ist) UND der Zustand an Eingang E 0.3 "1" ist.

Befindet sich der Strompfad aus dem Beispiel in einem aktiven MCR-Bereich:

Ist das MCR eingeschaltet, werden A 4.0 und A 4.1 entsprechend dem Signalzustand des Signalflusses wie oben beschrieben gesetzt.

Ist das MCR ausgeschaltet, werden A 4.0 und A 4.1 auf "0" zurückgesetzt, unabhängig vom Signalzustand des Signalflusses.

1.7 ---(#)--- Konnektor

Symbol

<Operand>

---(#)---

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, *L, D	Zugeordnetes Bit

* Ein Operand im Lokaldaten-Stack kann nur verwendet werden, wenn er in der Variablendeklarationstabelle im Bereich TEMP eines Codebausteins (FC, FB, OB) deklariert wurde.

Beschreibung

---(#)--- (Konnektor) ist ein zwischengeschaltetes Element mit Zuordnungsfunktion, das das aktuelle VKE (den Signalzustand des Signalflusses) an einem angegebenen **<Operanden>** speichert. Dieses Zuordnungselement speichert die Bitverknüpfung der letzten geöffneten Verzweigung vor dem Zuordnungselement. Bei Reihenschaltung mit anderen Elementen wird die Operation **---(#)---** wie ein Kontakt eingefügt. Das Element **---(#)---** darf nie an die Stromschiene angeschlossen werden bzw. es darf nie direkt hinter einer Verzweigung angeordnet werden und kann nicht als Abschluß eines Zweiges eingesetzt werden. Ein negiertes Element **---(#)---**kann mit dem Element ---|NOT|--- (Verknüpfungsergebnis invertieren) erstellt werden.

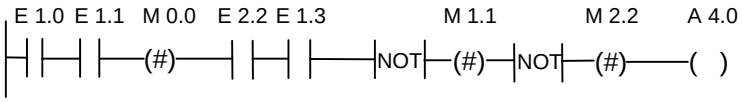
Abhängigkeit vom MCR (Master Control Relay)

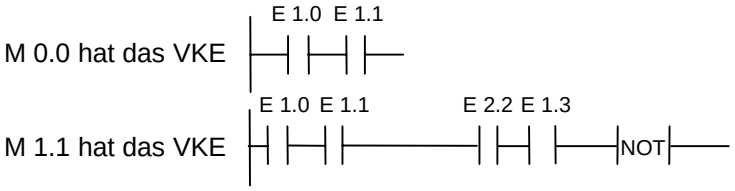
Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich ein Konnektor in einem aktiven MCR-Bereich befindet. Ist das MCR eingeschaltet und es fließt Strom zu einem Konnektor, dann wird das adressierte Bit auf den aktuellen Signalzustand des Signalflusses gesetzt. Ist das MCR ausgeschaltet, wird eine "0" an den angegebenen Operanden geschrieben, unabhängig vom Signalzustand des Signalflusses.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	-	1

Beispiel





M 2.2 hat das VKE der gesamten Bitverknüpfung

1.8 ---(R) Ausgang rücksetzen

Symbol

<Operand>

---(R)

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D, T, Z	Zurückgesetztes Bit

Beschreibung

---(R) (Ausgang rücksetzen) wird nur ausgeführt, wenn das VKE der vorherigen Operationen "1" ist (Signalfluß an der Spule). Fließt Strom zur Spule (VKE ist "1"), dann wird der angegebene **<Operand>** des Elements auf "0" gesetzt. Ein VKE von "0" (kein Signalfluß an der Spule) hat keine Auswirkungen, so daß der Signalzustand des angegebenen Operanden des Elements nicht verändert wird. Der **<Operand>** kann auch eine Zeit (T-Nr.) sein, deren Zeitwert auf "0" gesetzt wird, oder ein Zähler (Z-Nr.), dessen Zählwert auf "0" gesetzt wird.

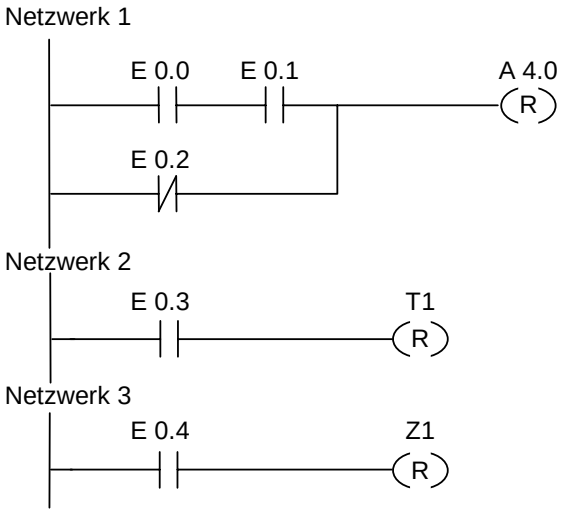
Abhängigkeit vom MCR (Master Control Relay)

Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich eine Spule in einem aktiven MCR-Bereich befindet. Ist das MCR eingeschaltet und es fließt Strom zu einer Spule, dann wird das adressierte Bit auf "0" gesetzt. Ist das MCR ausgeschaltet, wird der aktuelle Signalzustand des angegebenen Operanden des Elements nicht verändert, unabhängig vom Signalzustand des Signalflusses.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	-	0

Beispiel



Ausgang A 4.0 wird nur zurückgesetzt, wenn:

(der Zustand an Eingang E 0.0 UND an Eingang E 0.1 "1" ist) ODER der Zustand an Eingang E 0.2 "0" ist.

Die Zeit T1 wird nur zurückgesetzt, wenn:

der Signalzustand an Eingang E 0.3 "1" ist.

Der Zähler Z1 wird nur zurückgesetzt, wenn:

der Signalzustand an Eingang E 0.3 "1" ist.

Befindet sich der Strompfad des Beispiels in einem MCR-Bereich:

Wenn das MCR eingeschaltet ist, werden A 4.0, T1 und Z1 wie oben beschrieben zurückgesetzt.

Wenn das MCR ausgeschaltet ist, werden A 4.0, T1 und Z1 nicht verändert, unabhängig vom Signalzustand des VKE (Signalzustand des Signalflusses).

1.9 ---(S) Ausgang setzen

Symbol

<Operand>

---(S)

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Gesetztes Bit

Beschreibung

---(S) (Ausgang setzen) wird nur ausgeführt, wenn das VKE der vorherigen Operationen "1" ist (Signalfluß an der Spule). Ist das VKE "1", wird der angegebene **<Operand>** des Elements auf "1" gesetzt.

Ein VKE = 0 hat keine Auswirkungen, so daß der aktuelle Signalzustand des angegebenen Operanden des Elements nicht verändert wird.

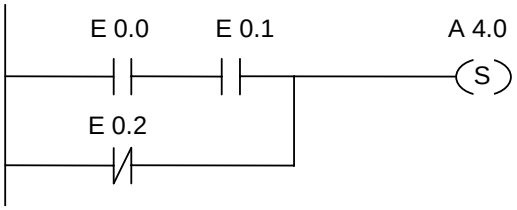
Abhängigkeit vom MCR (Master Control Relay)

Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich eine Spule in einem aktiven MCR-Bereich befindet. Ist das MCR eingeschaltet und es fließt Strom zu einer Spule, dann wird das adressierte Bit auf "1" gesetzt. Ist das MCR ausgeschaltet, wird der aktuelle Signalzustand des angegebenen Operanden des Elements nicht verändert, unabhängig vom Signalzustand des Signalflusses.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	-	0

Beispiel



Ausgang A 4.0 wird nur auf "1" gesetzt, wenn:

(der Zustand an Eingang E 0.0 UND an E 0.1 "1" ist) ODER der Zustand an Eingang E 0.2 "0" ist.

Ist das VKE "0", bleibt der Signalzustand von Ausgang A 4.0 gleich.

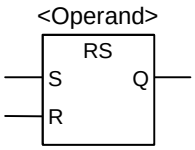
Befindet sich der Strompfad des Beispiels in einem MCR-Bereich:

Wenn das MCR eingeschaltet ist, wird A 4.0 wie oben beschrieben gesetzt.

Wenn das MCR ausgeschaltet ist, wird A 4.0 nicht verändert, unabhängig vom Signalzustand des VKE (Signalzustand des Signalflusses).

1.10 RS Flipflop rücksetzen setzen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Gesetztes oder zurückgesetztes Bit
S	BOOL	E, A, M, L, D	Setzen freigeben
R	BOOL	E, A, M, L, D	Rücksetzen freigeben
Q	BOOL	E, A, M, L, D	Signalzustand von <Operand>

Beschreibung

RS (Flipflop rücksetzen setzen) wird rückgesetzt, wenn am Eingang R der Zustand "1" ist und am Eingang S der Zustand "0" ist. Andernfalls, wenn am Eingang R der Zustand "0" und am Eingang S der Zustand "1" ist, wird das Flipflop gesetzt. Ist das VKE an beiden Eingängen "1", führt das RS-Flipflop an dem angegebenen **<Operand>** zunächst das Rücksetzen, dann das Setzen aus, so daß die Adresse für den Rest des Programmzyklus gesetzt bleibt.

Die Operationen S (Setzen) und R (Rücksetzen) werden nur ausgeführt, wenn das VKE = 1 ist. Ist das VKE = 0, werden diese Operationen nicht beeinflußt und der angegebene Operand wird nicht verändert.

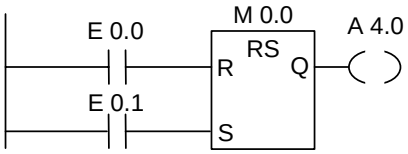
Abhängigkeit vom MCR (Master Control Relay)

Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich die Operation Flipflop rücksetzen setzen innerhalb eines aktiven MCR-Bereichs befindet. Ist das MCR eingeschaltet, dann wird das adressierte Bit auf "1" gesetzt bzw. auf "0" zurückgesetzt, wie oben beschrieben. Ist das MCR ausgeschaltet, wird der aktuelle Zustand des angegebenen Operanden unabhängig vom Zustand der Eingänge nicht verändert.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



Wenn der Zustand an Eingang E 0.0 "1" und an Eingang E 0.1 "0" ist, wird der Merker M 0.0 zurückgesetzt und A 4.0 ist "0". Andernfalls, wenn der Signalzustand an Eingang E 0.0 = 0 und an E 0.1 = 1 ist, wird der Merker M 0.0 gesetzt und A 4.0 ist "1". Wenn beide Signalzustände "0" sind, wird nichts verändert. Wenn beide Signalzustände "1" sind, dominiert aufgrund der Reihenfolge die Operation Setzen. M 0.0 wird gesetzt und A 4.0 ist "1".

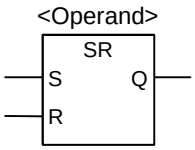
Befindet sich das Beispiel oben innerhalb eines aktiven MCR-Bereichs:

Wenn das MCR eingeschaltet ist, wird A 4.0 wie oben beschrieben gesetzt bzw. zurückgesetzt.

Wenn das MCR ausgeschaltet ist, wird A 4.0 nicht geändert, unabhängig vom Signalzustand der Eingänge.

1.11 SR Flipflop setzen rücksetzen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Gesetztes oder zurückgesetztes Bit
S	BOOL	E, A, M, L, D	Setzen freigeben
R	BOOL	E, A, M, L, D	Rücksetzen freigeben
Q	BOOL	E, A, M, L, D	Signalzustand von <Operand>

Beschreibung

SR (Flipflop setzen rücksetzen) wird gesetzt, wenn am Eingang S der Zustand "1" ist und am Eingang R der Zustand "0" ist. Andernfalls, wenn am Eingang S der Zustand "0" und am Eingang R der Zustand "1" ist, wird das Flipflop zurückgesetzt. Ist das VKE an beiden Eingängen "1", führt das SR-Flipflop an dem angegebenen **<Operanden>** zunächst das Setzen, dann das Rücksetzen aus, so daß die Adresse für den Rest des Programmzyklus zurückgesetzt bleibt.

Die Operationen S (Setzen) und R (Rücksetzen) werden nur ausgeführt, wenn das VKE = 1 ist. Ist das VKE = 0, werden diese Operationen nicht beeinflußt und der angegebene Operand wird nicht verändert.

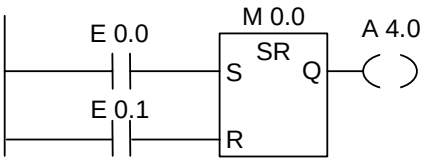
Abhängigkeit vom MCR (Master Control Relay)

Die Abhängigkeit vom MCR wird nur aktiviert, wenn sich die Operation Flipflop setzen rücksetzen innerhalb eines aktiven MCR-Bereichs befindet. Ist das MCR eingeschaltet, dann wird das adressierte Bit auf "1" gesetzt bzw. auf "0" zurückgesetzt, wie oben beschrieben. Ist das MCR ausgeschaltet, wird der aktuelle Zustand des angegebenen Operanden unabhängig vom Zustand der Eingänge nicht verändert.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



Wenn der Zustand an Eingang E 0.0 "1" und an Eingang E 0.1 "0" ist, wird der Merker M 0.0 gesetzt und A 4.0 ist "1". Andernfalls, wenn der Signalzustand an Eingang E 0.0 = 0 und an E 0.1 = 1 ist, wird der Merker M 0.0 zurückgesetzt und A 4.0 ist "0". Wenn beide Signalzustände "0" sind, wird nichts verändert. Wenn beide Signalzustände "1" sind, dominiert aufgrund der Reihenfolge die Operation Rücksetzen. M 0.0 wird zurückgesetzt und A 4.0 ist "0".

Befindet sich das Beispiel oben innerhalb eines aktiven MCR-Bereichs:

Wenn das MCR eingeschaltet ist, wird A 4.0 wie oben beschrieben gesetzt bzw. zurückgesetzt.

Wenn das MCR ausgeschaltet ist, wird A 4.0 nicht geändert, unabhängig vom Signalzustand der Eingänge.

1.12 ---(N)--- Flanke 1 -> 0 abfragen

Symbol

<Operand>

---(N)---

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Flankenmerker, speichert den vorherigen Signalzustand des VKE

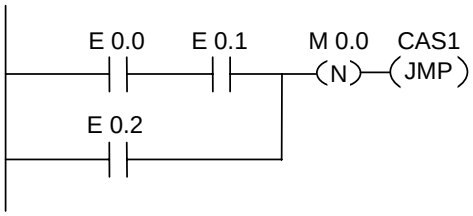
Beschreibung

---(N)--- (Flanke 1 -> 0 abfragen) erkennt einen Wechsel des Signalzustands im Operanden von "1" nach "0" und zeigt dies nach der Operation mit VKE = 1 an. Der aktuelle Signalzustand des VKE wird mit dem Signalzustand des Operanden, dem Flankenmerker, verglichen. Ist der Signalzustand des Operanden "1" und das VKE vor der Operation "0", so ist das VKE nach der Operation "1" (Impuls), in allen anderen Fällen "0". Das VKE vor der Operation wird im Operanden gespeichert.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	X	1

Beispiel



Der Flankenmerker M 0.0 speichert den Signalzustand des VKE aus der gesamten Bitverknüpfung.

Wenn der Signalzustand des VKE von "1" nach "0" wechselt, wird der Sprung zur Sprungmarke CAS1 ausgeführt.

1.13 ---(P)--- Flanke 0 -> 1 abfragen

Symbol

<Operand>

---(P)---

Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand>	BOOL	E, A, M, L, D	Flankenmerker, speichert den vorherigen Signalzustand des VKE

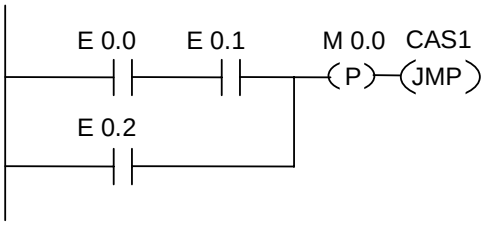
Beschreibung

---(P)--- (Flanke 0 -> 1 abfragen) erkennt einen Wechsel des Signalzustands im Operanden von "0" nach "1" und zeigt dies nach der Operation mit VKE = 1 an. Der aktuelle Signalzustand des VKE wird mit dem Signalzustand des Operanden, dem Flankenmerker, verglichen. Ist der Signalzustand des Operanden "0" und das VKE vor der Operation "1", so ist das VKE nach der Operation "1" (Impuls), in allen anderen Fällen "0". Das VKE vor der Operation wird im Operanden gespeichert.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	X	1

Beispiel



Der Flankenmerker M 0.0 speichert den Signalzustand des VKE aus der gesamten Bitverknüpfung. Wenn der Signalzustand des VKE von "0" nach "1" wechselt, wird der Sprung zur Sprungmarke CAS1 ausgeführt.

1.14 ---(SAVE) Verknüpfungsergebnis in BIE-Register laden

Symbol

---(SAVE)

Beschreibung

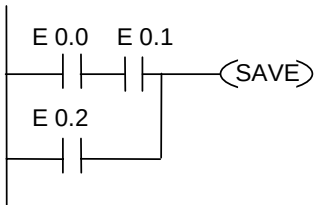
---(SAVE) (Verknüpfungsergebnis in BIE-Register laden) speichert das VKE im BIE-Bit des Statusworts. Das Erstabfragebit /ER wird dabei nicht zurückgesetzt. Aus diesem Grund wird bei einer UND-Verknüpfung im nächsten Netzwerk der Zustand des BIE-Bits mitverknüpft.

Die Verwendung von **SAVE** und eine nachfolgende Abfrage des BIE-Bits im gleichen Baustein oder in unterlagerten Bausteinen wird nicht empfohlen, da das BIE-Bit durch zahlreiche dazwischen liegende Operationen verändert werden kann. Sinnvoll ist der Einsatz der Operation **SAVE** vor Verlassen eines Baustein, da damit der ENO-Ausgang (=BIE-Bit) auf den Wert des VKE-Bits gesetzt wird und Sie daran eine Fehlerbehandlung des Bausteins anschließen können.

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	X	-	-	-	-	-	-	-	-

Beispiel

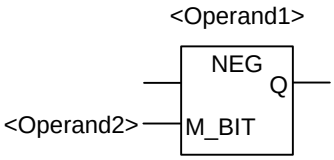


Der Status des Netzwerks (= VKE) wird im BIE-Bit gespeichert.

BIE Binärergebnisbit (Statuswort, Bit 8)

1.15 NEG Signalflanke 1 -> 0 abfragen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand1>	BOOL	E, A, M, L, D	Abgefragtes Signal
<Operand2>	BOOL	E, A, M, L, D	Flankenmerker M_BIT, speichert den vorherigen Signalzustand von <Operand1>
Q	BOOL	E, A, M, L, D	Signalwechselerkennung

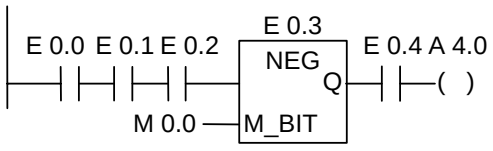
Beschreibung

NEG (Signalflanke 1 -> 0 abfragen) vergleicht den Signalzustand von **<Operand1>** mit dem Signalzustand der vorherigen Abfrage, der in **<Operand2>** gespeichert ist. Wenn der aktuelle Zustand des VKE "0" ist, und der vorherige Zustand "1" war (Erkennung einer fallenden Flanke), ist der Ausgang Q nach dieser Funktion "1", in allen anderen Fällen "0".

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	1	X	1

Beispiel

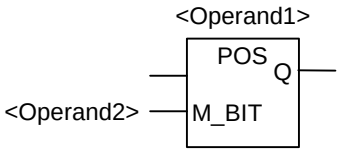


Der Ausgang A 4.0 ist "1", wenn:

(der Zustand an E 0.0 UND an E 0.1 UND an E 0.2 "1" ist) UND E 0.3 eine fallende Flanke hat UND der Zustand an E 0.4 "1" ist.

1.16 POS Signalflanke 0 -> 1 abfragen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
<Operand1>	BOOL	E, A, M, L, D	Abgefragtes Signal
<Operand2>	BOOL	E,A, M, L, D	Flankenmerker M_BIT, speichert den vorherigen Signalzustand von <Operand1>
Q	BOOL	E, A, M, L, D	Signalwechselerkennung

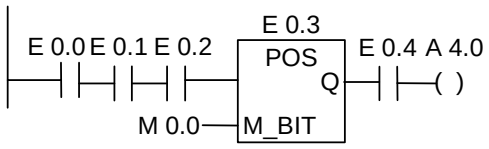
Beschreibung

POS (Signalflanke 0 -> 1 abfragen) vergleicht den Signalzustand von **<Operand1>** mit dem Signalzustand der vorherigen Abfrage, der in **<Operand2>** gespeichert ist. Wenn der aktuelle Zustand des VKE "1" ist, und der vorherige Zustand "0" war (Erkennung einer steigenden Flanke), ist der Ausgang Q nach dieser Operation "1", in allen anderen Fällen "0".

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	1	X	1

Beispiel



Der Ausgang A 4.0 ist "1", wenn:

(der Zustand an E 0.0 UND an E 0.1 UND an E 0.2 "1" ist) UND E 0.3 eine steigende Flanke hat UND der Zustand an E 0.4 "1" ist.

1.17 Peripherie direkt lesen

Beschreibung

Für die Funktion **Peripherie direkt lesen** müssen Sie ein Netzwerk erstellen (siehe Beispiel).

In zeitkritischen Anwendungen kann es sein, daß der aktuelle Zustand eines digitalen Eingangs häufiger gelesen werden muß als im Normalfall (einmal pro Zyklus). Die Operation "Peripherie direkt lesen" erhält den Zustand des digitalen Eingangs von der Eingabebaugruppe zu dem Zeitpunkt, zu dem der entsprechende Strompfad gelesen wird. Andernfalls müssen Sie bis zum nächsten OB1-Zyklus warten, wenn der Speicherbereich der Eingänge mit dem Zustand des Speicherbereichs der Peripherie aktualisiert ist.

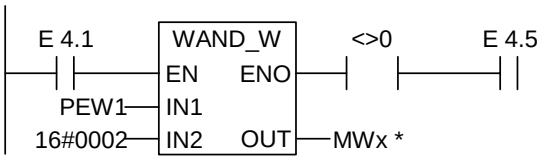
Wenn Sie den Eingang (oder mehrere Eingänge) direkt aus der Eingabebaugruppe lesen möchten, verwenden Sie den Speicherbereich Peripherie Eingänge (PE) im Gegensatz zu dem Speicherbereich der Eingänge (E). Der Speicherbereich der Peripherie kann als Byte, Wort oder Doppelwort gelesen werden. Deshalb kann ein einzelner digitaler Eingang nicht über einen Kontakt (Bit) gelesen werden.

Bedingtes Übertragen von Spannung in Abhängigkeit von dem Zustand eines direkten Eingangs:

1. Das Wort des Speicherbereichs PE, das die relevanten Daten enthält, wird von der CPU gelesen.
2. Das Wort des Speicherbereichs PE wird dann mit einer Konstanten durch UND verknüpft, die ein Ergebnis ungleich Null zuläßt, wenn das Eingangsbit eingeschaltet ist ("1").
3. Es wird auf die Bedingung ungleich Null geprüft.

Beispiel

KOP-Netzwerk mit der Operation **Peripherie direkt lesen** für Eingang E 1.1.



* MWx muß angegeben werden, um das Netzwerk speichern zu können. X steht für eine beliebige, zulässige Nummer.

PEW1	0000000000101010
W#16#0002	0000000000000010
Ergebnis	0000000000000010

In diesem Beispiel ist der direkte Eingang E 1.1 in Reihe geschaltet mit den Eingängen E 4.1 und E 4.5.

Das Wort PEW1 enthält den direkten Zustand von E 1.1. PEW1 wird mit W#16#0002 durch UND verknüpft. Das Ergebnis ist ungleich Null, wenn E 1.1 (zweites Bit) in PB1 wahr ist ("1"). Der Kontakt U<>0 überträgt die Spannung, wenn das Ergebnis der Operation WAND_W ungleich Null ist.

1.18 Peripherie direkt schreiben

Beschreibung

Für die Funktion **Peripherie direkt schreiben** müssen Sie ein Netzwerk erstellen (siehe Beispiel).

In zeitkritischen Anwendungen kann es sein, daß der aktuelle Zustand eines digitalen Ausgangs häufiger an eine Ausgabebaugruppe übertragen werden muß als im Normalfall (einmal am Ende eines Zyklus von OB1). Die Operation "Peripherie direkt schreiben" aktualisiert den Zustand eines digitalen Ausgangs in der Ausgabebaugruppe zu dem Zeitpunkt, zu dem der entsprechende Strompfad beschrieben wird. Andernfalls müssen Sie bis zum Ende des OB1-Zyklus warten, wenn der Speicherbereich der Peripherie mit dem Zustand des Speicherbereichs der Ausgänge aktualisiert wird.

Wenn Sie den Ausgang (oder mehrere Ausgänge) direkt aktualisieren möchten, verwenden Sie den Speicherbereich Peripherie Ausgänge (PA) im Gegensatz zu dem Speicherbereich der Ausgänge (A). Der Speicherbereich der Peripherie Ausgänge kann als Byte, Wort und Doppelwort beschrieben werden. Deshalb kann ein einzelner digitaler Ausgang nicht über eine Spule aktualisiert werden. Um den Zustand eines digitalen Ausgangs direkt in eine Ausgabebaugruppe zu schreiben, wird ein Byte, Wort oder Doppelwort des Speicherbereichs der Ausgänge A, das das relevante Bit enthält, bedingt in den entsprechenden PA-Speicher (in die Operanden der direkten Ausgabebaugruppe) kopiert.

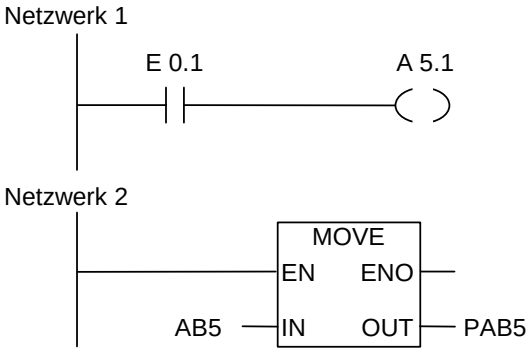
Warnung

- Weil das vollständige Byte des Speicherbereichs A in die Ausgabebaugruppe geschrieben wird, werden alle Ausgangsbits in dem Byte, das aktualisiert wird, ebenfalls geändert, wenn die Operation ausgeführt wird.
- Hat ein Ausgangsbit Zwischenzustände (1/0), die während des Programms auftreten und nicht in die Ausgabebaugruppen gesendet werden dürfen, dann kann die Operation Peripherie direkt schreiben gefährliche Zustände hervorrufen (Übergangsimpulse an Ausgängen).
- Als allgemeine Regel gilt für den Aufbau, daß eine externe Ausgabebaugruppe in einem Programm nur einmal als Spule adressiert werden darf. Wenn Sie diese Regel beachten, vermeiden Sie die meisten Probleme, die die Operation "Peripherie direkt schreiben" hervorrufen kann.

Beispiel

KOP-Netzwerk mit der Operation **Peripherie direkt schreiben** und der digitalen Ausgabebaugruppe 5, Kanal 1.

Die Zustände der Bits des adressierten Ausgangsbytes (AB5) werden entweder aktualisiert oder nicht verändert. A 5.1 wird im Netzwerk 1 der Signalzustand von E 0.1 zugewiesen. AB5 wird in den entsprechenden direkten Speicherbereich der Peripherie Ausgänge (PAB5) kopiert.



In diesem Beispiel ist A 5.1 das geforderte Ausgangsbit.

Das Byte PAB5 enthält den Zustand des Ausgangsbits A 5.1.

Die anderen Bits in PAB5 werden durch das Kopieren mit der Operation MOVE auch aktualisiert.

2 Vergleicher

2.1 Vergleichsoperationen Übersicht

Beschreibung

Verglichen werden die Eingänge IN1 und IN2 entsprechend der folgenden Vergleichsarten:

== IN1 ist gleich IN2
<> IN1 ist ungleich IN2
> IN1 ist größer als IN2
< IN1 ist kleiner als IN2
>= IN1 ist größer als oder gleich IN2
<= IN1 ist kleiner als oder gleich IN2

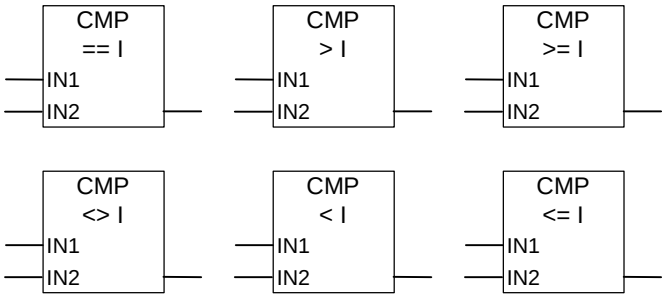
Ist der Vergleich wahr, ist das VKE der Operation "1". Das VKE wird mit dem VKE des gesamten Strompfads durch UND verknüpft, sofern das Vergleichselement in Reihe geschaltet ist, bzw. durch ODER, sofern die Box parallel geschaltet ist.

Folgende Vergleichsoperationen stehen Ihnen zur Verfügung:

- CMP ? I Ganze Zahlen vergleichen (16 Bit)
- CMP ? D Ganze Zahlen vergleichen (32 Bit)
- CMP ? R Gleitpunktzahlen vergleichen

2.2 CMP ? I Ganze Zahlen vergleichen (16 Bit)

Symbole



Parameter	Datentyp	Speicherbereich	Beschreibung
Boxeingang	BOOL	E, A, M, L, D	Ergebnis der vorherigen Verknüpfung
Boxausgang	BOOL	E, A, M, L, D	Ergebnis des Vergleichs, es wird nur dann weiter-verarbeitet wenn VKE am Boxeingang = 1.
IN1	INT	E, A, M, L, D oder Konstante	Erster Vergleichswert
IN2	INT	E, A, M, L, D oder Konstante	Zweiter Vergleichswert

Beschreibung

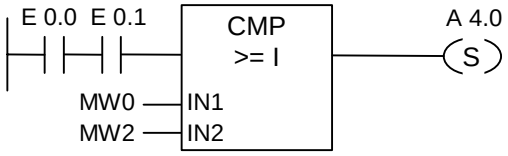
CMP ? I (Ganze Zahlen vergleichen, 16 Bit) kann wie ein normaler Kontakt verwendet werden. Die Box kann an den Stellen eingesetzt werden, an denen auch ein normaler Kontakt angeordnet werden kann. IN1 und IN2 werden nach der von Ihnen gewählten Vergleichsart verglichen.

Ist der Vergleich wahr, ist das VKE der Operation "1". Das VKE wird mit dem VKE des gesamten Strompfads durch UND verknüpft, sofern das Vergleichselement in Reihe geschaltet ist, bzw. durch ODER, sofern die Box parallel geschaltet ist.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	0	-	0	X	X	1

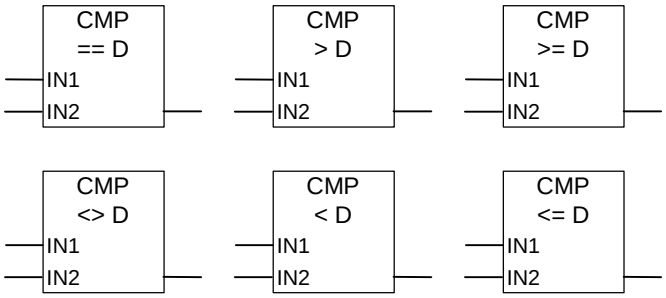
Beispiel



A 4.0 wird gesetzt, wenn E 0.0 UND E 0.1 = 1 sind UND wenn MW0 >= MW2 ist.

2.3 CMP ? D Ganze Zahlen vergleichen (32 Bit)

Symbole



Parameter	Datentyp	Speicherbereich	Beschreibung
Boxeingang	BOOL	E, A, M, L, D	Ergebnis der vorherigen Verknüpfung
Boxausgang	BOOL	E, A, M, L, D	Ergebnis des Vergleichs, es wird nur dann weiter-verarbeitet wenn VKE am Boxeingang = 1.
IN1	DINT	E, A, M, L, D oder Konstante	Erster Vergleichswert
IN2	DINT	E, A, M, L, D oder Konstante	Zweiter Vergleichswert

Beschreibung

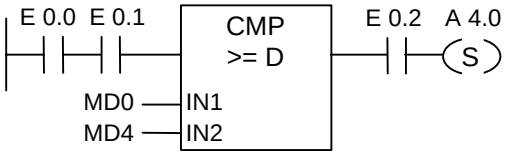
CMP ? D (Ganze Zahlen vergleichen, 32 Bit) kann als normaler Kontakt verwendet werden. Die Box kann an den Stellen eingesetzt werden, an denen auch ein normaler Kontakt angeordnet werden kann. IN1 und IN2 werden nach der von Ihnen gewählten Vergleichsart verglichen.

Ist der Vergleich wahr, ist das VKE der Operation "1". Das VKE wird mit dem VKE des gesamten Strompfads durch UND verknüpft, sofern das Vergleichselement in Reihe geschaltet ist, bzw. durch ODER, sofern die Box parallel geschaltet ist.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	0	-	0	X	X	1

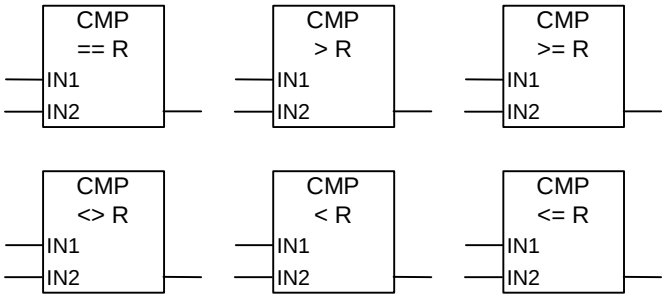
Beispiel



A 4.0 wird gesetzt, wenn E 0.0 UND E 0.1 = 1 sind UND wenn MD0 >= MD4 ist UND wenn E 0.2 = 1 ist.

2.4 CMP ? R Gleitpunktzahlen vergleichen

Symbole



Parameter	Datentyp	Speicherbereich	Beschreibung
Boxeingang	BOOL	E, A, M, L, D	Ergebnis der vorherigen Verknüpfung
Boxausgang	BOOL	E, A, M, L, D	Ergebnis des Vergleichs, es wird nur dann weiterverarbeitet wenn VKE am Boxeingang = 1.
IN1	REAL	E, A, M, L, D oder Konstante	Erster Vergleichswert
IN2	REAL	E, A, M, L, D oder Konstante	Zweiter Vergleichswert

Beschreibung

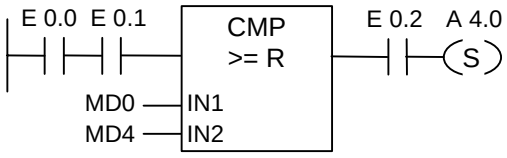
CMP ? R (Gleitpunktzahlen vergleichen) kann wie ein normaler Kontakt verwendet werden. Die Box kann an den Stellen eingesetzt werden, an denen auch ein normaler Kontakt angeordnet werden kann. IN1 und IN2 werden nach der von Ihnen gewählten Vergleichsart verglichen.

Ist der Vergleich wahr, ist das VKE der Operation "1". Das VKE wird mit dem VKE des gesamten Strompfads durch UND verknüpft, sofern das Vergleichselement in Reihe geschaltet ist, bzw. durch ODER, sofern die Box parallel geschaltet ist.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

Beispiel



A 4.0 wird gesetzt, wenn E 0.0 UND E 0.1 = 1 sind UND wenn MD0 >= MD4 ist UND wenn E 0.2 = 1 ist.

3 Umwandler

3.1 Umwandlungsoperationen Übersicht

Beschreibung

Die Umwandlungsoperationen lesen den Inhalt des Parameters IN und wandeln diesen um oder kehren das Vorzeichen um. Das Ergebnis kann am Parameter OUT abgefragt werden.

Folgende Operationen stehen Ihnen zur Umwandlung zur Verfügung:

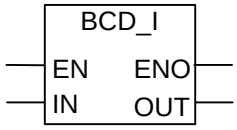
- BCD_I BCD-Zahl in 16-Bit-Ganzzahl wandeln
- I_BCD 16-Bit-Ganzzahl in BCD-Zahl wandeln
- BCD_DI BCD-Zahl in 32-Bit-Ganzzahl wandeln
- I_DI 16-Bit-Ganzzahl in 32-Bit-Ganzzahl wandeln
- DI_BCD 32-Bit-Ganzzahl in BCD-Zahl wandeln
- DI_R 32-Bit-Ganzzahl in Gleitpunktzahl wandeln

- INV_I 1er Komplement zu 16-Bit-Ganzzahl erzeugen
- INV_DI 1er Komplement zu 32-Bit-Ganzzahl erzeugen
- NEG_I 2er Komplement zu 16-Bit-Ganzzahl erzeugen
- NEG_DI 2er Komplement zu 32-Bit-Ganzzahl erzeugen

- NEG_R Vorzeichen einer Gleitpunktzahl wechseln
- ROUND Zahl runden
- TRUNC Ganze Zahl erzeugen
- CEIL Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen
- FLOOR Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen

3.2 BCD_I BCD-Zahl in 16-Bit-Ganzzahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	WORD	E, A, M, L, D	BCD-Zahl
OUT	INT	E, A, M, L, D	Ganzzahliger Wert (16 Bit) der BCD-Zahl

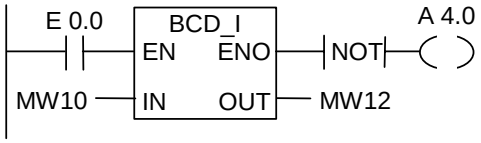
Beschreibung

BCD_I (BCD-Zahl in 16-Bit-Ganzzahl wandeln) liest den Inhalt des Parameters IN als dreistellige, BCD-Zahl (+/- 999) und wandelt diese Zahl in einen ganzzahligen Wert (16 Bit) um. Das ganzzahlige Ergebnis kann am Parameter OUT abgefragt werden. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

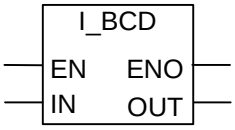
Beispiel



Wenn der Eingang E 0.0 = 1 ist, dann wird der Inhalt von MW10 als dreistellige, BCD-Zahl gelesen und in eine Ganzzahl (16 Bit) umgewandelt. Das Ergebnis wird in MW12 abgelegt.
Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wurde (ENO = EN = 0).

3.3 I_BCD 16-Bit-Ganzzahl in BCD-Zahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	INT	E, A, M, L, D	Ganzzahl (16 Bit)
OUT	WORD	E, A, M, L, D	BCD-Wert der Ganzzahl (16 Bit)

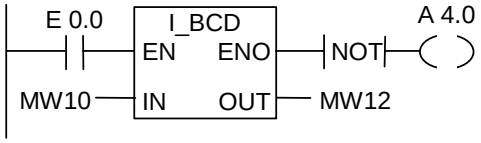
Beschreibung

I_BCD (16-Bit-Ganzzahl in BCD-Zahl wandeln) liest den Inhalt des Parameters IN als ganzzahligen Wert (16 Bit) und wandelt diesen Wert in eine dreistellige, BCD-Zahl (+/- 999) um. Das Ergebnis kann am Parameter OUT abgefragt werden. Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	X	X	0	X	X	1

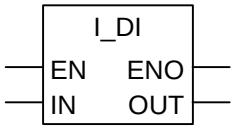
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MW10 als Ganzzahl (16 Bit) gelesen und in eine dreistellige, BCD-Zahl umgewandelt. Das Ergebnis wird in MW12 abgelegt. Ausgang A 4.0 ist "1", wenn ein Überlauf auftritt oder die Anweisung nicht bearbeitet wird (E 0.0 = 0).

3.4 I_DI 16-Bit-Ganzzahl in 32-Bit-Ganzzahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	INT	E, A, M, L, D	Ganzzahliger Wert (16 Bit), der umgewandelt werden soll
OUT	DINT	E, A, M, L, D	Ergebnis: Ganzzahl (32 Bit)

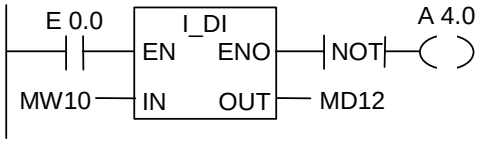
Beschreibung

I_DI (16-Bit-Ganzzahl in 32-Bit-Ganzzahl wandeln) liest den Inhalt des Parameters IN als Ganzzahl (16 Bit) und wandelt diese Zahl in eine Ganzzahl (32 Bit) um. Das Ergebnis kann am Parameter OUT abgefragt werden. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

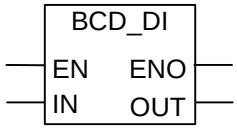
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MW10 als Ganzzahl (16 Bit) gelesen und in eine Ganzzahl (32 Bit) umgewandelt. Das Ergebnis wird in MD12 abgelegt. Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

3.5 BCD_DI BCD-Zahl in 32-Bit-Ganzzahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DWORD	E, A, M, L, D	BCD-Zahl
OUT	DINT	E, A, M, L, D	Ganzzahliger Wert (32 Bit) der BCD-Zahl

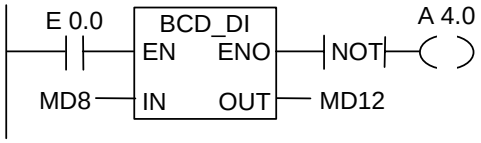
Beschreibung

BCD_DI (BCD-Zahl in 32-Bit-Ganzzahl wandeln) liest den Inhalt des Parameters IN als siebenstellige, BCD-Zahl (+/- 9999999) und wandelt diese Zahl in einen ganzzahligen Wert (32 Bit) um. Das ganzzahlige Ergebnis kann am Parameter OUT abgefragt werden. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

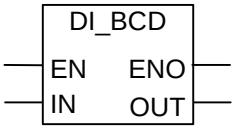
Beispiel



Wenn der Eingang E 0.0 = 1 ist, dann wird der Inhalt von MD8 als siebenstellige, BCD-Zahl gelesen und in eine Ganzzahl (32 Bit) umgewandelt. Das Ergebnis wird in MD12 abgelegt.
Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wurde (ENO = EN = 0).

3.6 DI_BCD 32-Bit-Ganzzahl in BCD-Zahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DINT	E, A, M, L, D	Ganzzahl (32 Bit)
OUT	DWORD	E, A, M, L, D	BCD-Wert der Ganzzahl (32 Bit)

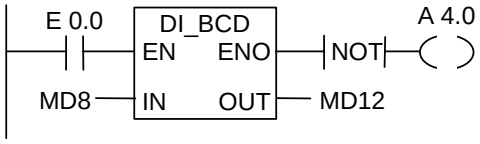
Beschreibung

DI_BCD (32-Bit-Ganzzahl in BCD-Zahl wandeln) liest den Inhalt des Parameters IN als ganzzahligen Wert (32 Bit) und wandelt diesen Wert in eine siebenstellige, BCD-Zahl (+/- 9999999) um. Das Ergebnis kann am Parameter OUT abgefragt werden. Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	X	X	0	X	X	1

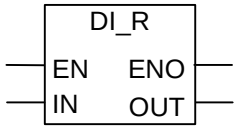
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Ganzzahl (32 Bit) gelesen und in eine siebenstellige, BCD-Zahl umgewandelt. Das Ergebnis wird in MD12 abgelegt. Ausgang A 4.0 ist "1", wenn ein Überlauf auftritt oder die Anweisung nicht bearbeitet wird (E 0.0 = 0).

3.7 DI_R 32-Bit-Ganzzahl in Gleitpunktzahl wandeln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DINT	E, A, M, L, D	Ganzzahl (32 Bit)
OUT	REAL	E, A, M, L, D	Gleitpunktzahl

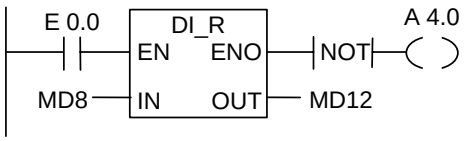
Beschreibung

DI_R (32-Bit-Ganzzahl in Gleitpunktzahl wandeln) liest den Inhalt des Parameters IN als ganzzahligen Wert (32 Bit) und wandelt diesen Wert in eine Gleitpunktzahl um. Das Ergebnis kann am Parameter OUT abgefragt werden. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

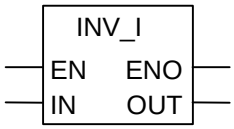
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Ganzzahl (32 Bit) gelesen und in eine Gleitpunktzahl umgewandelt. Das Ergebnis wird in MD12 abgelegt. Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

3.8 INV_I 1er Komplement zu 16-Bit-Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	INT	E, A, M, L, D	Ganzzahliger Eingangswert (16 Bit)
OUT	INT	E, A, M, L, D	Einerkomplement der Ganzzahl (16 Bit) von IN

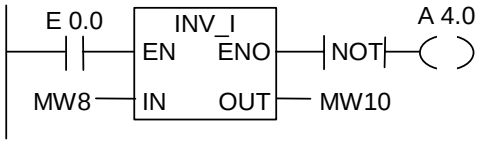
Beschreibung

INV_I (1er Komplement zu 16-Bit-Ganzzahl erzeugen) liest den Inhalt des Parameters IN und verknüpft den Wert mit der Hexadezimalschablone W#16#FFFF durch EXKLUSIV ODER. Diese Operation kehrt den Zustand jedes einzelnen Bits um. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

Beispiel



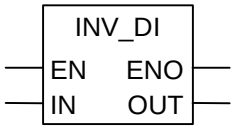
Ist E 0.0 = 1, dann wird der Zustand jedes einzelnen Bits von MW8 umgekehrt:

MW8 = 01000001 10000001 -> MW10 = 10111110 01111110

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

3.9 INV_DI 1er Komplement zu 32-Bit-Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DINT	E, A, M, L, D	Ganzzahliger Eingangswert (32 Bit)
OUT	DINT	E, A, M, L, D	Einerkomplement der Ganzzahl (32 Bit) von IN

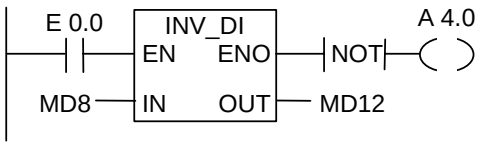
Beschreibung

INV_DI (1er Komplement zu 32-Bit-Ganzzahl erzeugen) liest den Inhalt des Parameters IN und verknüpft den Wert mit der Hexadezimalschablone W#16#FFFF FFFF durch EXKLUSIV ODER. Diese Operation kehrt den Zustand jedes einzelnen Bits um. ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

Beispiel



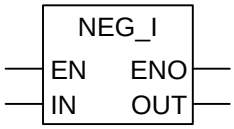
Ist E 0.0 = 1, dann wird der Zustand jedes einzelnen Bits von MD8 umgekehrt:

MD8 = F0FF FFF0 -> MD12 = 0F00 000F

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

3.10 NEG_I 2er Komplement zu 16-Bit-Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	INT	E, A, M, L, D	Ganzzahliger Eingangswert (16 Bit)
OUT	INT	E, A, M, L, D	Zweierkomplement der Ganzzahl (16 Bit) von IN

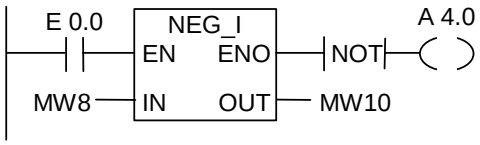
Beschreibung

NEG_I (2er Komplement zu 16-Bit-Ganzzahl erzeugen) liest den Inhalt des Parameters IN und führt die Operation Zweierkomplement aus. Die Operation wechselt das Vorzeichen (zum Beispiel von einem positiven Wert in einen negativen Wert). ENO hat immer den gleichen Signalzustand wie EN, mit folgender Ausnahme: Wenn der Signalzustand von EN = 1 ist und ein Überlauf auftritt, ist der Signalzustand von ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

Beispiel



Wenn E 0.0 = 1 ist, dann wird der Wert von MW8 mit dem umgekehrten Vorzeichen vom Parameter OUT an MW10 ausgegeben:

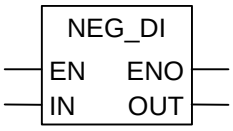
MW8 = + 10 -> MW10 = - 10

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

Wenn der Signalzustand von EN = 1 ist und ein Überlauf auftritt, ist der Signalzustand von ENO = 0.

3.11 NEG_DI 2er Komplement zu 32-Bit-Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DINT	E, A, M, L, D	Ganzzahliger Eingangswert (32 Bit)
OUT	DINT	E, A, M, L, D	Zweierkomplement der Ganzzahl (32 Bit) von IN

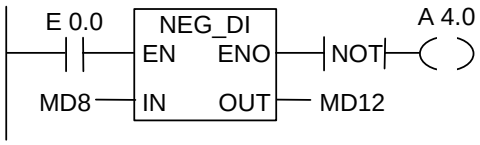
Beschreibung

NEG_DI (2er Komplement zu 32-Bit-Ganzzahl erzeugen) liest den Inhalt des Parameters IN und führt die Operation Zweierkomplement aus. Die Operation wechselt das Vorzeichen (zum Beispiel von einem positiven Wert in einen negativen Wert). ENO hat immer den gleichen Signalzustand wie EN, mit folgender Ausnahme: wenn der Signalzustand von EN = 1 ist und ein Überlauf auftritt, ist der Signalzustand von ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

Beispiel



Wenn E 0.0 = 1 ist, dann wird der Wert von MD8 mit dem umgekehrten Vorzeichen vom Parameter OUT an MD12 ausgegeben.

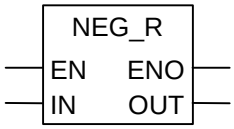
MD8 = + 1000 -> MD12 = - 1000

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

Wenn der Signalzustand von EN = 1 ist und ein Überlauf auftritt, ist der Signalzustand von ENO = 0.

3.12 NEG_R Vorzeichen einer Gleitpunktzahl wechseln

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Gleitpunktzahl von IN mit negierten Vorzeichen

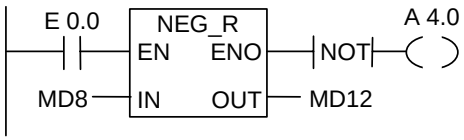
Beschreibung

NEG_R (Vorzeichen einer Gleitpunktzahl wechseln) liest den Inhalt des Parameters IN und wechselt das Vorzeichen. Die Operation entspricht einer Multiplikation mit (-1). Die Operation wechselt das Vorzeichen (Beispiel: ein positiver Wert wird zum negativen Wert). ENO hat immer den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	-	-	0	X	X	1

Beispiel



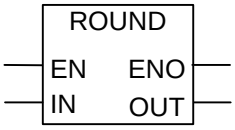
Ist E 0.0 = 1, dann wird der Wert von MD8 mit umgekehrtem Vorzeichen vom Parameter OUT an MD12 ausgegeben:

MD8 = + 6,234 -> MD12 = - 6,234

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

3.13 ROUND Zahl runden

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Wert, der gerundet werden soll
OUT	DINT	E, A, M, L, D	IN, zur nächsten ganzen Zahl gerundet

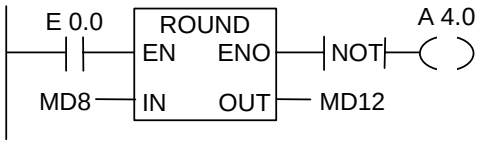
Beschreibung

ROUND (Zahl auf/abrunden) liest den Inhalt des Parameters IN als Gleitpunktzahl und wandelt diese in eine Ganzzahl (32 Bit) um. Das Ergebnis ist die nächstgelegene Ganzzahl ("Auf/Abrunden"). Falls die Gleitpunktzahl in der Mitte zwischen zwei Ganzzahlen liegt, wird die gerade Zahl zurückgeliefert. Das Ergebnis wird im Parameter OUT abgelegt. Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	X	X	0	X	X	1

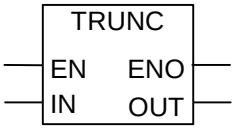
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Gleitpunktzahl gelesen und in die nächstgelegene Ganzzahl (32 Bit) umgewandelt. Das Ergebnis dieser Funktion "Auf/Abrunden" wird in MD12 abgelegt.
Ausgang A 4.0 ist "1", wenn ein Überlauf auftritt oder die Anweisung nicht bearbeitet wird (E 0.0 = 0).

3.14 TRUNC Ganze Zahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Gleitpunktzahl, die umgewandelt werden soll
OUT	DINT	E, A, M, L, D	Ganzzahliger Teil des Werts von IN

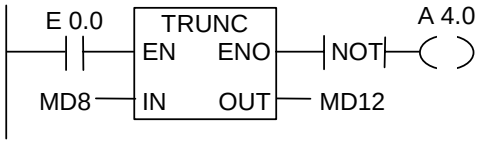
Beschreibung

TRUNC (Ganze Zahl erzeugen) liest den Inhalt des Parameters IN als Gleitpunktzahl und wandelt diesen Wert in eine Ganzzahl (32 Bit) um. Das Ergebnis ist der ganzzahlige Anteil der Gleitpunktzahl, der vom Parameter OUT ausgegeben wird. Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	X	X	0	X	X	1

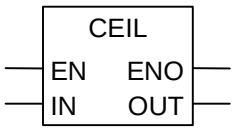
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Gleitpunktzahl gelesen und in eine Ganzzahl (32 Bit) umgewandelt. Das Ergebnis ist der ganzzahlige Teil der Gleitpunktzahl, der in MD12 gespeichert wird.
Ausgang A 4.0 ist "1", wenn ein Überlauf eintritt oder die Anweisung nicht bearbeitet wird (E 0.0 = 0).

3.15 CEIL Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Gleitpunktzahl, die umgewandelt werden soll
OUT	DINT	E, A, M, L, D	Kleinste Ganzzahl (32 Bit), die größer ist als die Gleitpunktzahl

Beschreibung

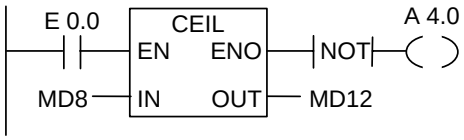
CEIL (Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen) liest den Inhalt des Parameters IN als Gleitpunktzahl und wandelt diese in eine Ganzzahl (32 Bit) um. Das Ergebnis ist die kleinste Ganzzahl, die größer ist als die Gleitpunktzahl ("Aufrunden"). Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt *	X	-	-	X	X	0	X	X	1
schreibt**	0	-	-	-	-	0	0	0	1

* Operation wird ausgeführt (EN = 1)
** Operation wird nicht ausgeführt (EN = 0)

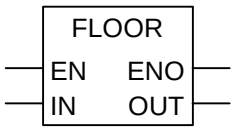
Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Gleitpunktzahl gelesen und diese mit der Funktion "Aufrunden" in eine Ganzzahl (32 Bit) umgewandelt. Das Ergebnis wird in MD12 abgelegt. Ausgang A 4.0 ist "1", wenn ein Überlauf auftritt oder die Anweisung nicht bearbeitet wird (E0.0 = 0).

3.16 FLOOR Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Gleitpunktzahl, die umgewandelt werden soll
OUT	DINT	E, A, M, L, D	Größte Ganzzahl (32 Bit), die kleiner als die Gleitpunktzahl ist

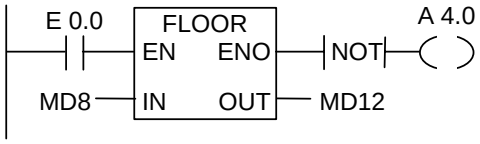
Beschreibung

FLOOR (Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen) liest den Inhalt des Parameters IN als Gleitpunktzahl und wandelt diese in eine Ganzzahl (32 Bit) um. Das Ergebnis ist die größte Ganzzahl, die kleiner ist als die Gleitpunktzahl ("Abrunden"). Tritt ein Überlauf auf, ist ENO = 0.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	-	-	X	X	0	X	X	1

Beispiel



Ist E 0.0 = 1, dann wird der Inhalt von MD8 als Gleitpunktzahl gelesen und diese mit der Funktion "Abrunden" in eine Ganzzahl (32 Bit) umgewandelt. Das Ergebnis wird in MD12 abgelegt. Ausgang A 4.0 ist "1", wenn ein Überlauf auftritt oder die Anweisung nicht bearbeitet wird (E 0.0 = 0).

4 Zähler

4.1 Zähloperationen Übersicht

Speicherbereich

Zähler haben einen eigenen reservierten Speicherbereich in Ihrer CPU. Dieser Speicherbereich reserviert ein Wort von 16 Bit für jeden Zähler. Das Programmieren mit KOP unterstützt 256 Zähler. Die Anzahl der Zähler ist von der CPU abhängig.

Zählwert

Die Bits 0 bis 9 des Zählerworts enthalten den Zählwert binär-codiert. Wenn der Zähler gesetzt wird, wird der von Ihnen festgelegte Wert vom Akkumulator in den Zähler übertragen. Der Bereich des Zählwerts liegt zwischen 0 und 999.

Sie können den Zählwert innerhalb dieses Bereichs mit folgenden Zähloperationen verändern:

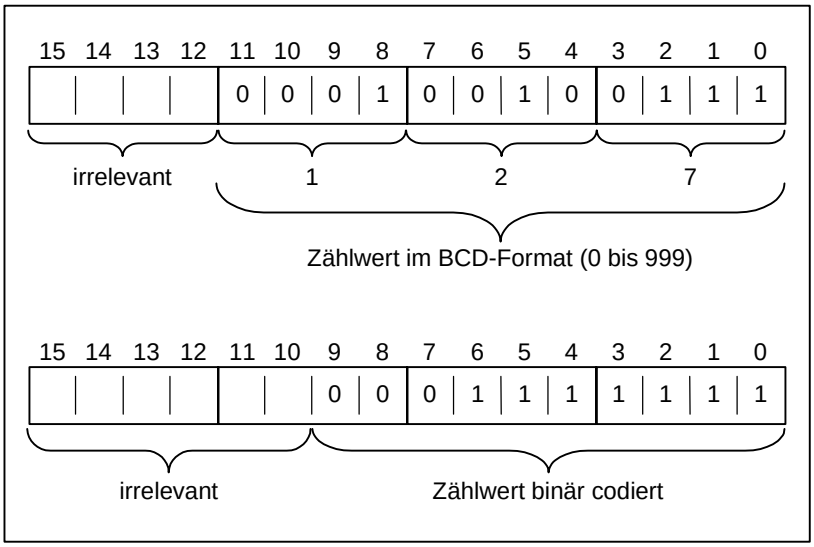
- ZÄHLER Parametrieren und vorwärts-/rückwärtszählen
- Z_RUECK Parametrieren und rückwärtszählen
- Z_VORW Parametrieren und vorwärtszählen
- ---(SZ) Zähleranfangswert setzen
- ---(ZV) Vorwärtszählen
- ---(ZR) Rückwärtszählen

Bit-Konfiguration

Ein Zähler wird auf einen bestimmten Wert gesetzt, indem Sie eine Zahl zwischen 0 und 999 im BCD-Format als Zählwert laden, z. B. C# 127.

Die Bits 0 bis 11 des Zählers enthalten den Zählwert im BCD-Format, d. h. jede Gruppe von 4 Bits enthält jeweils den Binärcode für einen Dezimalwert.

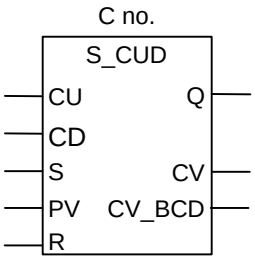
Das folgende Bild zeigt den Inhalt des Zählers, nachdem Sie den Zählwert 127 geladen haben, und den Inhalt des Zählerworts nach dem Setzen des Zählers.



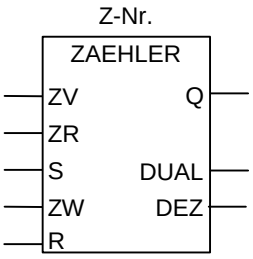
4.2 ZÄHLER Parametrieren und vorwärts-/rückwärtszählen

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
C no.	Z-Nr.	COUNTER	Z	Nummer des Zählers, Anzahl der Zähler ist von der CPU abhängig
CU	ZV	BOOL	E, A, M, L, D	Vorwärtszähleingang
CD	ZR	BOOL	E, A, M, L, D	Rückwärtszähleingang
S	S	BOOL	E, A, M, L, D	Eingang zum Voreinstellen des Zählers
PV	ZW	WORD	E, A, M, L, D oder Konstante	Zählwert eingegeben als C#<Wert> im Bereich zwischen 0 und 999
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
CV	DUAL	WORD	E, A, M, L, D	Rest-Zählwert, Hexadezimalzahl
CV_BCD	DEZ	WORD	E, A, M, L, D	Rest-Zählwert, BCD-Zahl
Q	Q	BOOL	E, A, M, L, D	Status des Zählers

Beschreibung

ZÄHLER (Parametrieren und vorwärts-/rückwärtszählen) wird bei steigender Flanke am Eingang S mit dem Wert des Eingangs ZW voreingestellt. Liegt am Eingang R eine 1, wird der Zähler zurückgesetzt und der Zählwert ist 0.

Der Zähler wird um "1" erhöht, wenn der Signalzustand am Eingang ZV von "0" auf "1" wechselt und der Wert des Zählers kleiner ist als "999".

Der Zähler wird um "1" vermindert, wenn am Eingang ZR eine steigende Flanke anliegt und der Wert des Zählers größer als "0" ist.

Liegt an beiden Zähleingängen eine steigende Flanke an, werden beide Operationen ausgeführt und der Zählwert bleibt unverändert.

Wird der Zähler gesetzt und ist an den Eingängen ZV/ZR das VKE = 1, so zählt der Zähler entsprechend im nächsten Zyklus, auch wenn kein Flankenwechsel gegeben war.

Der Signalzustand am Ausgang Q ist "1", wenn der Zählwert größer als Null ist, und "0", wenn der Zählwert gleich Null ist.

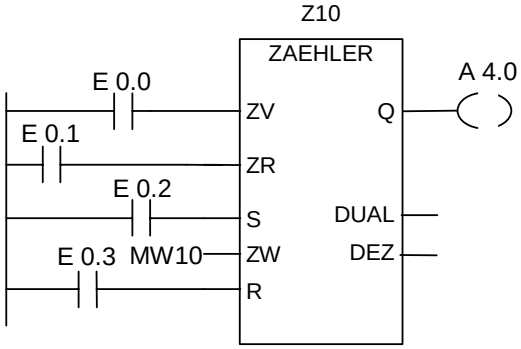
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Hinweis

Verwenden Sie einen Zähler nur an einer Stelle im Programm, um Zählfehler zu vermeiden.

Beispiel

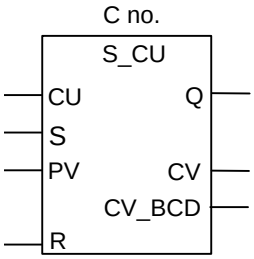


Wechselt E 0.2 von "0" auf "1", wird der Zähler mit dem Wert aus MW10 voreingestellt. Wechselt der Signalzustand an E 0.0 von "0" auf "1", wird der Wert des Zählers Z10 um "1" erhöht, es sei denn, der Wert von Z10 ist gleich "999". Wechselt E 0.1 von "0" auf "1", wird Z10 um "1" verringert, es sei denn, der Wert von Z10 ist gleich Null. A 4.0 ist "1", wenn Z10 ungleich Null ist.

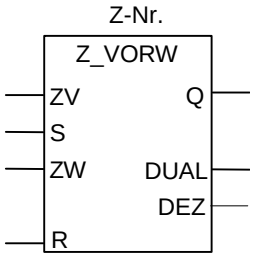
4.3 Z_VORW Parametrieren und vorwärtszählen

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
C no.	Z-Nr.	COUNTER	Z	Nummer des Zählers, Anzahl der Zähler ist von der CPU abhängig
CU	ZV	BOOL	E, A, M, L, D	Vorwärtszähleingang
S	S	BOOL	E, A, M, L, D	Eingang zum Voreinstellen des Zählers
PV	ZW	WORD	E, A, M, L, D oder Konstante	Zählwert eingegeben als C#<Wert> im Bereich zwischen 0 und 999
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
CV	DUAL	WORD	E, A, M, L, D	Rest-Zählwert, Hexadezimalzahl
CV_BCD	DEZ	WORD	E, A, M, L, D	Rest-Zählwert, BCD-Zahl
Q	Q	BOOL	E, A, M, L, D	Status des Zählers

Beschreibung

Z_VORW (Parametrieren und vorwärtszählen) wird mit dem Wert von Eingang ZW voreingestellt, wenn an Eingang S eine steigende Flanke anliegt.

Der Zähler wird zurückgesetzt, wenn eine 1 am Eingang R anliegt. Der Zählwert ist dann Null.

Der Zähler wird um "1" erhöht, wenn der Signalzustand am Eingang ZV von "0" auf "1" wechselt und der Wert des Zählers kleiner ist als "999".

Wird der Zähler gesetzt und ist am Eingang ZV das VKE = 1, so zählt der Zähler entsprechend im nächsten Zyklus, auch wenn kein Flankenwechsel gegeben war.

Der Signalzustand am Ausgang Q ist "1", wenn der Zählwert größer als Null ist, und "0", wenn der Zählwert gleich Null ist.

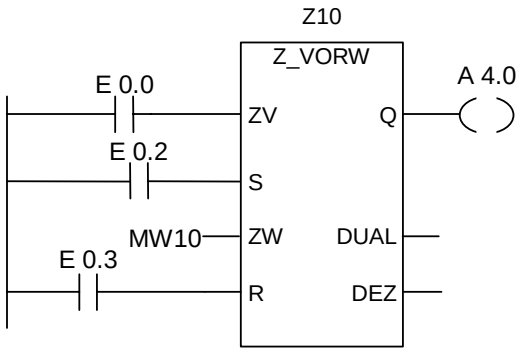
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Hinweis

Verwenden Sie einen Zähler nur an einer Stelle im Programm, um Zählfehler zu vermeiden.

Beispiel

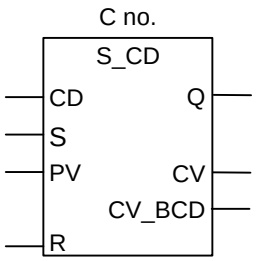


Wechselt E 0.2 von "0" auf "1", dann wird der Zähler mit dem Wert aus MW10 voreingestellt. Wechselt der Signalzustand an E 0.0 von "0" auf "1", dann wird der Wert des Zählers Z10 um "1" erhöht, es sei denn der Wert von Z10 ist gleich "999". A 4.0 ist "1", wenn Z10 ungleich Null ist.

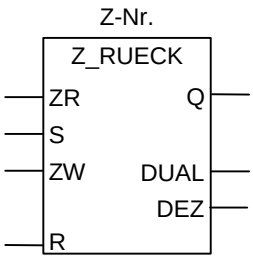
4.4 Z_RUECK Parametrieren und rückwärtszählen

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
C no.	Z-Nr.	COUNTER	Z	Nummer des Zählers, Anzahl der Zähler ist von der CPU abhängig
CD	ZR	BOOL	E, A, M, L, D	Rückwärtszähleingang
S	S	BOOL	E, A, M, L, D	Eingang zum Voreinstellen des Zählers
PV	ZW	WORD	E, A, M, L, D oder Konstante	Zählwert eingegeben als C#<Wert> im Bereich zwischen 0 und 999
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
CV	DUAL	WORD	E, A, M, L, D	Rest-Zählwert, Hexadezimalzahl
CV_BCD	DEZ	WORD	E, A, M, L, D	Rest-Zählwert, BCD-Zahl
Q	Q	BOOL	E, A, M, L, D	Status des Zählers

Beschreibung

Z_RUECK (Parametrieren und Rückwärtszählen) wird mit dem Wert von Eingang ZW voreingestellt, wenn an Eingang S eine steigende Flanke anliegt.

Der Zähler wird zurückgesetzt, wenn eine 1 am Eingang R anliegt. Der Zählwert ist dann Null.

Der Zähler wird um "1" verringert, wenn der Signalzustand am Eingang ZR von "0" auf "1" wechselt und der Wert des Zählers größer als Null ist.

Wird der Zähler gesetzt und ist am Eingang ZR das VKE = 1, so zählt der Zähler entsprechend im nächsten Zyklus, auch wenn kein Flankenwechsel gegeben war.

Der Signalzustand am Ausgang Q ist "1", wenn der Zählwert größer als Null ist, und "0", wenn der Zählwert gleich Null ist.

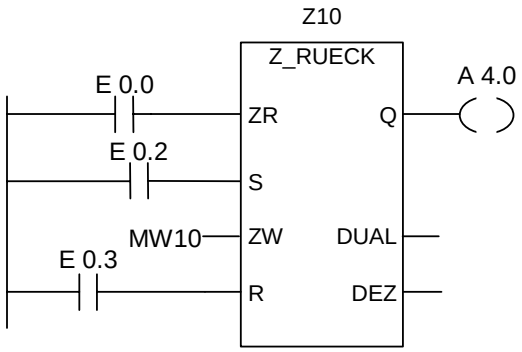
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Hinweis

Verwenden Sie einen Zähler nur an einer Stelle im Programm, um Zählfehler zu vermeiden.

Beispiel



Wechselt E 0.2 von "0" auf "1", dann wird der Zähler mit dem Wert aus MW10 voreingestellt. Wechselt der Signalzustand an E 0.0 von "0" auf "1", dann wird der Wert des Zählers Z10 um "1" verringert, es sei denn der Wert von Z10 ist gleich "0". A 4.0 ist "1", wenn Z10 ungleich Null ist.

4.5 ---(SZ) Zähleranfangswert setzen

Symbol

Englisch	**Deutsch**
<C-Nr.>	<Z-Nr.>
---(SC)	**---(SZ)**
<voreingestellter Wert>	<voreingestellter Wert>

Parameter Englisch	**Parameter Deutsch**	**Datentyp**	**Speicher-bereich**	**Beschreibung**
<C-Nr.>	<Z-Nr.>	COUNTER	Z	Nummer des Zählers, der mit einem Wert voreingestellt werden soll.
<vorein-gestellter Wert>	<vorein-gestellter Wert>	WORD	E, A, M, L, D oder Konstante	Der Wert zum Voreinstellen kann zwischen 0 und 999 liegen. Bei Eingabe einer Konstanten muß vor dem Wert C# stehen.

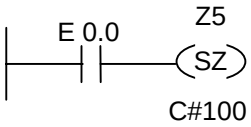
Beschreibung

---(SZ) (Zähleranfangswert setzen) wird nur bei einer steigenden Flanke im VKE ausgeführt. Dann wird der voreingestellte Wert in den angegebenen Zähler übertragen.

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	X	-	0

Beispiel



Der Zähler Z5 wird mit dem Wert "100" voreingestellt, wenn am Eingang E 0.0 eine steigende Flanke auftritt (Wechsel von "0" auf "1"). Ist keine steigende Flanke vorhanden, wird der Wert des Zählers Z5 nicht verändert.

4.6 ---(ZV) Vorwärtszählen

Symbol

Englisch	Deutsch
<C-Nr.>	<Z-Nr.>
---(CU)	**---(ZV)**

Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
<C-Nr.>	<Z-Nr.>	COUNTER	Z	Nummer des Zählers, dessen Wert erhöht werden soll. Die Anzahl der Zähler ist von der CPU abhängig.

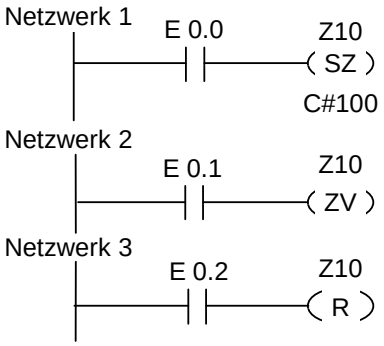
Beschreibung

---(ZV) (Vorwärtszählen) inkrementiert den Wert des angegebenen Zählers um "1", wenn im VKE eine steigende Flanke vorliegt und der Wert des Zählers kleiner als "999" ist. Ist keine steigende Flanke vorhanden oder der Zähler hat bereits den Wert "999", wird der Wert des Zählers nicht geändert.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	X	-	0

Beispiel



Wechselt der Signalzustand von E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird der voreingestellte Wert von "100" in den Zähler Z10 geladen.

Wechselt der Signalzustand von E 0.1 von "0" auf "1" (steigende Flanke im VKE), dann wird der Zählwert des Zählers Z10 um "1" erhöht, es sei denn der Zählwert ist gleich "999". Ist keine steigende Flanke im VKE vorhanden, wird der Wert des Zählers Z10 nicht geändert.

Ist der Signalzustand von E 0.2 = 1, dann wird der Zähler auf "0" zurückgesetzt.

4.7 ---(ZR) Rückwärtszählen

Symbol

Englisch	**Deutsch**
<C-Nr.>	<Z-Nr.>
---(CD)	**---(ZR)**

Parameter Englisch	**Parameter Deutsch**	**Datentyp**	**Speicher-bereich**	**Beschreibung**
<C-Nr.>	<Z-Nr.>	COUNTER	Z	Nummer des Zählers, dessen Wert vermindert werden soll. Die Anzahl der Zähler ist von der CPU abhängig.

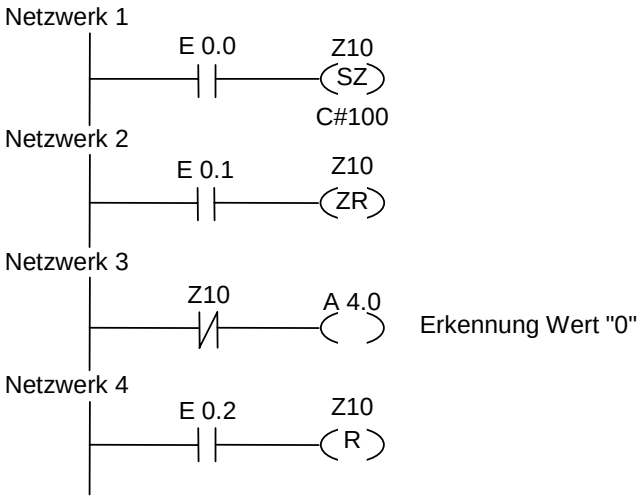
Beschreibung

---(ZR) (Rückwärtszählen) dekrementiert den Wert des angegebenen Zählers um "1", wenn im VKE eine steigende Flanke vorliegt und der Wert des Zählers größer als "0" ist. Ist keine steigende Flanke vorhanden oder der Zähler hat bereits den Wert "0", wird der Wert des Zählers nicht geändert.

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand von E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird der voreingestellte Wert von "100" in den Zähler Z10 geladen.

Wechselt der Signalzustand von E 0.1 von "0" auf "1" (steigende Flanke im VKE), dann wird der Zählwert des Zählers Z10 um "1" verringert, es sei denn der Zählwert ist gleich "0". Ist keine steigende Flanke im VKE vorhanden, wird der Wert des Zählers Z10 nicht geändert.

Ist der Zählwert gleich Null, dann wird A 4.0 eingeschaltet.

Ist der Signalzustand von E 0.2 = "1", dann wird der Zähler auf "0" zurückgesetzt.

5 DB-Aufruf

5.1 ---(OPN) Datenbaustein öffnen

Symbol

<DB-Nr.> oder <DI-Nr.>

---(OPN)

Parameter	Datentyp	Speicherbereich	Beschreibung
<DB-Nr.> <DI-Nr.>	BLOCK_DB	DB, DI	Nummer des DB/DI; Bereich ist von der CPU abhängig

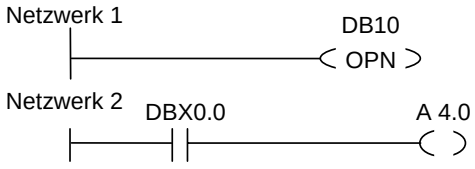
Beschreibung

---(OPN) (Datenbaustein öffnen) öffnet einen Datenbaustein (Global-DB oder Instanz-DB). Bei der Operation **---(OPN)** handelt es sich um einen absoluten Aufruf eines Datenbausteins. Die Nummer des Datenbausteins wird in das DB- bzw. DI-Register übertragen. Die darauffolgenden DB- und DI-Befehle greifen in Abhängigkeit der Registerinhalte auf die entsprechenden Bausteine zu.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	-	-	-	-

Beispiel



Datenbaustein 10 (DB 10) wird geöffnet. Die Adresse des Kontakts (DBX0.0) bezieht sich auf das Bit Null des Datenbytes Null des aktuellen Datensatzes, der sich in DB 10 befindet. Der Signalzustand dieses Bits wird Ausgang A 4.0 zugewiesen.

6 Sprünge

6.1 Sprungoperationen Übersicht

Beschreibung

Sprungoperation können Sie in allen Codebausteinen verwenden, z. B. in Organisationsbausteinen (OBs), Funktionsbausteinen (FBs) und Funktionen (FCs).

Folgende Sprungoperationen stehen Ihnen zur Verfügung:

- ---(JMP)--- Springe im Baustein absolut
- ---(JMP)--- Springe im Baustein wenn 1 (bedingt)
- ---(JMPN)--- Springe im Baustein wenn 0 (bedingt)

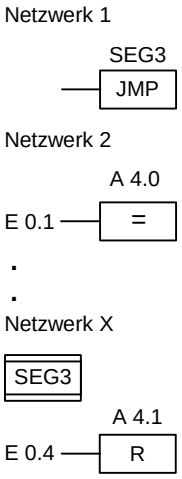
Sprungmarke als Operand

Der Operand einer Sprungoperation ist eine Sprungmarke. Sie gibt das Ziel an, zu dem das Programm springen soll.

Die Sprungmarke geben Sie über der Box JMP ein. Die Sprungmarke besteht aus max. 4 Zeichen. Das erste Zeichen muß ein Buchstabe sein, die anderen Zeichen können Buchstaben oder Zahlen sein (z. B. SEG3).

Sprungmarke als Ziel

Die Zielsprungmarke muß am Anfang eines Netzwerks stehen. Sie geben die Zielsprungmarke ein, indem Sie aus der KOP-Auswahlbox LABEL wählen. Es erscheint eine leere Box, in die Sie den Namen der Sprungmarke eingeben.



6.2 ---(JMP)--- Springe im Baustein absolut

Symbol

<Sprungmarke>

---(JMP)

Beschreibung

---(JMP) (Springe im Baustein absolut) funktioniert als absoluter Sprung, wenn zwischen linker Stromschiene und der Operation kein weiteres KOP-Element steht (siehe Beispiel).

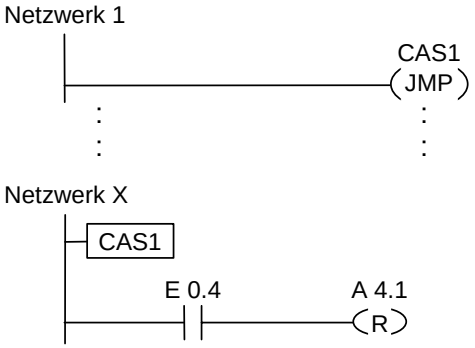
Zu jedem ---(JMP) muß auch ein Ziel (LABEL) vorhanden sein.

Die Operationen zwischen der Sprungoperation und der Sprungmarke werden nicht ausgeführt.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	-	-	-	-

Beispiel



Der Sprung wird immer ausgeführt, und die Operationen zwischen der Sprungoperation und der Sprungmarke werden übersprungen.

6.3 ---(JMP) Springe im Baustein wenn 1

Symbol

<Sprungmarke>

---(JMP)

Beschreibung

---(JMP) (Im Baustein springen wenn 1) funktioniert als bedingter Sprung, wenn das VKE der vorhergehenden Verknüpfung "1" ist.

Zu jedem ---(JMP) muß auch ein Ziel (LABEL) vorhanden sein.

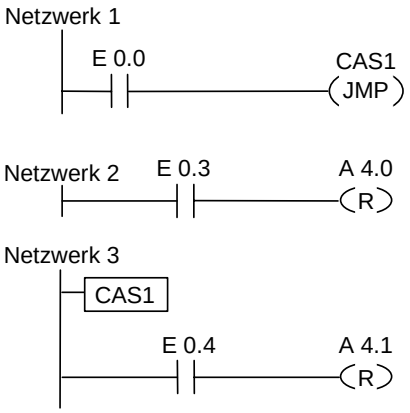
Die Operationen zwischen der Sprungoperation und der Sprungmarke werden nicht ausgeführt!

Wird ein bedingter Sprung nicht ausgeführt, wechselt das VKE nach der Sprungoperation auf "1".

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	1	1	0

Beispiel



Wenn E 0.0 = 1 ist, dann wird der Sprung zur Sprungmarke CAS1 ausgeführt. Wegen des Sprungs wird die Operation Ausgang rücksetzen an A 4.0 nicht ausgeführt, auch wenn E 0.3 = 1 ist.

6.4 ---(JMPN) Springe im Baustein wenn 0 (bedingt)

Symbol

<Sprungmarke>

---(JMPN)

Beschreibung

---(JMPN) (Springe im Baustein wenn 0) funktioniert als bedingter Sprung, wenn das VKE der vorhergehenden Verknüpfung "0" ist.

Zu jedem ---(JMPN) muß auch ein Ziel (LABEL) vorhanden sein.

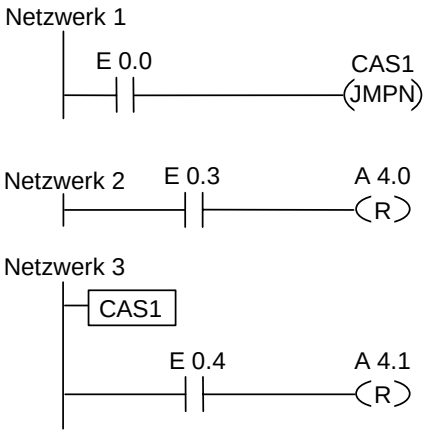
Die Operationen zwischen der Sprungoperation und der Sprungmarke werden nicht ausgeführt!

Wird ein bedingter Sprung nicht ausgeführt, wechselt das VKE nach der Sprungoperation auf "1".

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	1	1	0

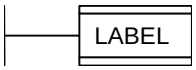
Beispiel



Ist E 0.0 = 0, dann wird der Sprung zur Sprungmarke CAS1 ausgeführt. Wegen des Sprungs wird die Operation Ausgang rücksetzen an A 4.0 nicht ausgeführt, auch wenn E 0.3 = 1 ist.

6.5 LABEL Sprungmarke

Symbol

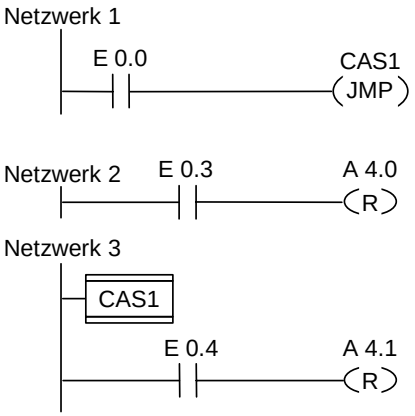


Beschreibung

LABEL kennzeichnet das Ziel einer Sprungoperation. Es besteht aus 4 Zeichen - erstes Zeichen: Buchstabe, Rest: Buchstabe oder alphanumerisch, z.B. CAS1.

Zu jedem ---(JMP) oder ---(JMPN) muß auch eine Sprungmarke (LABEL) vorhanden sein.

Beispiel



Wenn E 0.0 = 1 ist, dann wird der Sprung zur Sprungmarke CAS1 ausgeführt. Wegen des Sprungs wird die Operation Ausgang rücksetzen an A 4.0 nicht ausgeführt, auch wenn E 0.3 = 1 ist.

7 Festpunkt-Funktionen

7.1 Festpunkt-Funktionen Übersicht

Beschreibung

Mit den Festpunkt-Funktionen können Sie folgende Operationen mit **zwei Ganzzahlen** (16 Bit, 32 Bit) ausführen:

- ADD_I Ganze Zahlen addieren (16 Bit)
- SUB_I Ganze Zahlen subtrahieren (16 Bit)
- MUL_I Ganze Zahlen multiplizieren (16 Bit)
- DIV_I Ganze Zahlen dividieren (16 Bit)

- ADD_DI Ganze Zahlen addieren (32 Bit)
- SUB_DI Ganze Zahlen subtrahieren (32 Bit)
- MUL_DI Ganze Zahlen multiplizieren (32 Bit)
- DIV_DI Ganze Zahlen dividieren (32 Bit)
- MOD_DI Divisionsrest gewinnen (32 Bit)

7.2 Auswerten der Bits im Statuswort bei Festpunkt-Funktionen

Beschreibung

Die Festpunkt-Funktionen beeinflussen die Bits A1, A0, OV und OS im Statuswort.

Die folgenden Tabellen zeigen den Signalzustand der Bits des Statusworts für die Ergebnisse von Operationen mit Festpunktzahlen (16 Bit, 32 Bit).

Gültiger Bereich	A1	A0	OV	OS
0 (Null)	0	0	0	*
16 Bit: -32 768 <= Ergebnis < 0 (negative Zahl) 32 Bit: -2 147 483 648 <=Ergebnis < 0 (negative Zahl)	0	1	0	*
16 Bit: 32 767 >= Ergebnis > 0 (positive Zahl) 32 Bit: 2 147 483 647 >= Ergebnis > 0 (positive Zahl)	1	0	0	*

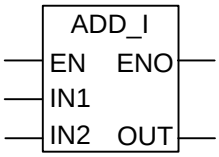
* Das OS-Bit wird vom Ergebnis der Operation nicht beeinflußt.

Ungültiger Bereich	A1	A0	OV	OS
Unterschreitung bei Addition 16 Bit: Ergebnis = -65536 32 Bit: Ergebnis = -4 294 967 296	0	0	1	1
Unterschreitung bei Multiplikation 16 Bit: Ergebnis < -32 768 (negative Zahl) 32 Bit: Ergebnis < -2 147 483 648 (negative Zahl)	0	1	1	1
Überlauf bei Addition, Subtraktion 16 Bit: Ergebnis > 32 767 (positive Zahl) 32 Bit: Ergebnis > 2 147 483 647 (positive Zahl)	0	1	1	1
Überlauf bei Multiplikation, Division 16 Bit: Ergebnis > 32 767 (positive Zahl) 32 Bit: Ergebnis > 2 147 483 647 (positive Zahl)	1	0	1	1
Unterschreitung bei Addition, Subtraktion 16 Bit: Ergebnis < -32 768 (negative Zahl) 32 Bit: Ergebnis < -2 147 483 648 (negative Zahl)	1	0	1	1
Division durch 0	1	1	1	1

Operation	A1	A0	OV	OS
+D: Ergebnis = -4 294 967 296	0	0	1	1
/D oder MOD: Division durch 0	1	1	1	1

7.3 ADD_I Ganze Zahlen addieren (16 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	INT	E, A, M, L, D oder Konstante	Erster Wert der Addition
IN2	INT	E, A, M, L, D oder Konstante	Zweiter Wert der Addition
OUT	INT	E, A, M, L, D	Ergebnis der Addition

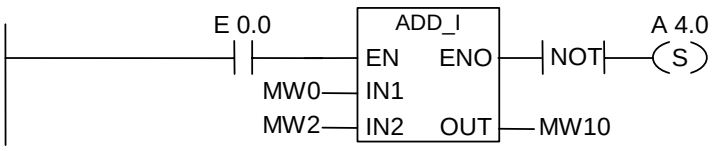
Beschreibung

ADD_I (Ganze Zahlen addieren, 16 Bit) addiert IN1 und IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

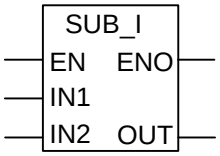
Beispiel



Die Box ADD_I wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Addition MW0 + MW2 wird von MW10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.4 SUB_I Ganze Zahlen subtrahieren (16 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	INT	E, A, M, L, D oder Konstante	Erster Wert der Subtraktion
IN2	INT	E, A, M, L, D oder Konstante	Wert, der subtrahiert werden soll
OUT	INT	E, A, M, L, D	Ergebnis der Subtraktion

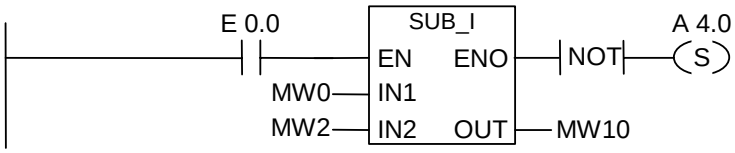
Beschreibung

SUB_I (Ganze Zahlen subtrahieren, 16 Bit) subtrahiert IN2 von IN1, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

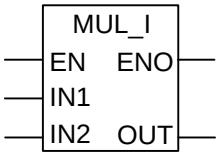
Beispiel



Die Box SUB_I wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Subtraktion MW0 - MW2 wird von MW10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.5 MUL_I Ganze Zahlen multiplizieren (16 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	INT	E, A, M, L, D oder Konstante	Erster Wert der Multiplikation
IN2	INT	E, A, M, L, D oder Konstante	Zweiter Wert der Multiplikation
OUT	INT	E, A, M, L, D	Ergebnis der Multiplikation

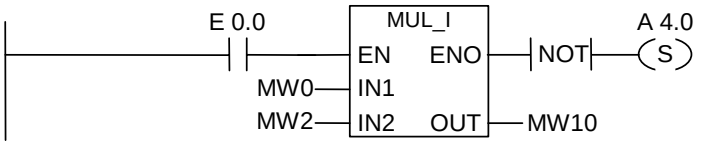
Beschreibung

MUL_I (Ganze Zahlen multiplizieren, 16 Bit) multipliziert IN1 und IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

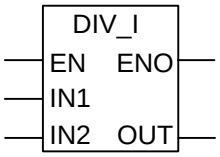
Beispiel



Die Box MUL_I wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Multiplikation MW0 x MW2 wird von MW10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.6 DIV_I Ganze Zahlen dividieren (16 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	INT	E, A, M, L, D oder Konstante	Dividend
IN2	INT	E, A, M, L, D oder Konstante	Divisor
OUT	INT	E, A, M, L, D	Ergebnis der Division

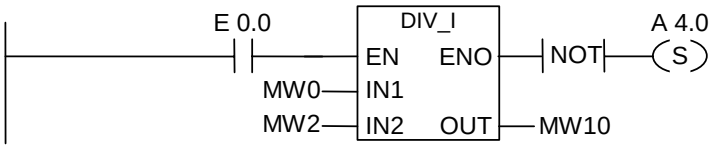
Beschreibung

DIV_I (Ganze Zahlen dividieren, 16 Bit) dividiert IN1 durch IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

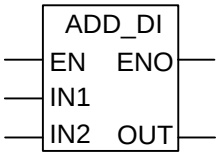
Beispiel



Die Box DIV_I wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Division MW0 durch MW2 wird von MW10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (16 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.7 ADD_DI Ganze Zahlen addieren (32 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DINT	E, A, M, L, D oder Konstante	Erster Wert der Addition
IN2	DINT	E, A, M, L, D oder Konstante	Zweiter Wert der Addition
OUT	DINT	E, A, M, L, D	Ergebnis der Addition

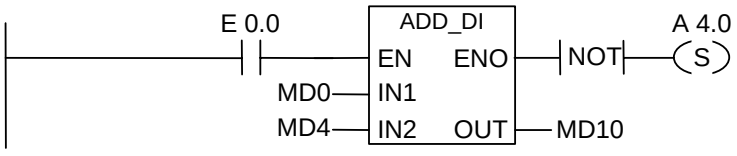
Beschreibung

ADD_DI (Ganze Zahlen addieren, 32 Bit) addiert IN1 und IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

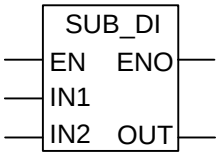
Beispiel



Die Box ADD_DI wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Addition MD0 + MD4 wird von MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.8 SUB_DI Ganze Zahlen subtrahieren (32 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DINT	E, A, M, L, D oder Konstante	Erster Wert der Subtraktion
IN2	DINT	E, A, M, L, D oder Konstante	Wert, der subtrahiert werden soll
OUT	DINT	E, A, M, L, D	Ergebnis der Subtraktion

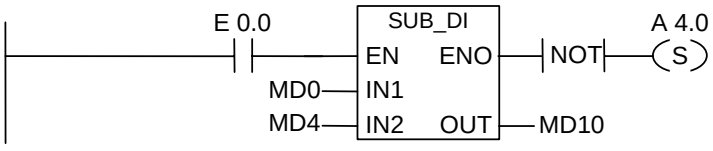
Beschreibung

SUB_DI (Ganze Zahlen subtrahieren, 32 Bit) subtrahiert IN2 von IN1, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

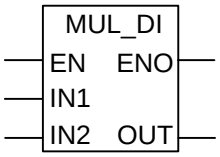
Beispiel



Die Box SUB_DI wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Subtraktion MD0 - MD4 wird von MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.9 MUL_DI Ganze Zahlen multiplizieren (32 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DINT	E, A, M, L, D oder Konstante	Erster Wert der Multiplikation
IN2	DINT	E, A, M, L, D oder Konstante	Zweiter Wert der Multiplikation
OUT	DINT	E, A, M, L, D	Ergebnis der Multiplikation

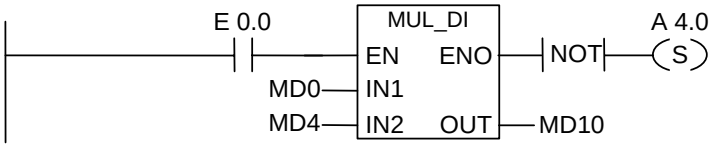
Beschreibung

MUL_DI (Ganze Zahlen multiplizieren, 32 Bit) multipliziert die Eingänge IN1 und IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

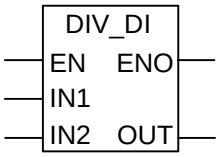
Beispiel



Die Box MUL_DI wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Multiplikation MD0 X MD4 wird von MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.10 DIV_DI Ganze Zahlen dividieren (32 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DINT	E, A, M, L, D oder Konstante	Dividend
IN2	DINT	E, A, M, L, D oder Konstante	Divisor
OUT	DINT	E, A, M, L, D	ganzzahliges Ergebnis der Division

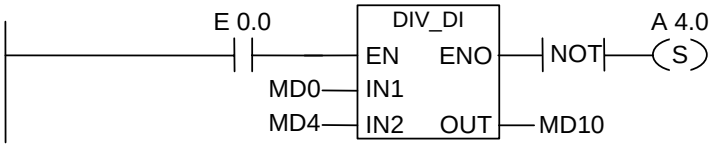
Beschreibung

DIV_DI (Ganze Zahlen dividieren, 32 Bit) dividiert IN1 durch IN2, wenn der Signalzustand am Freigabeeingang EN "1" ist. Das Ergebnis (ganzzahliger Anteil) kann an OUT abgefragt werden. Das Element Ganze Zahlen dividieren (32 Bit) erzeugt keinen Divisionsrest. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

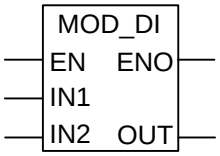
Beispiel



Die Box DIV_DI wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Division MD0 durch MD4 wird von MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

7.11 MOD_DI Divisionsrest gewinnen (32 Bit)

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DINT	E, A, M, L, D oder Konstante	Dividend
IN2	DINT	E, A, M, L, D oder Konstante	Divisor
OUT	DINT	E, A, M, L, D	Divisionsrest

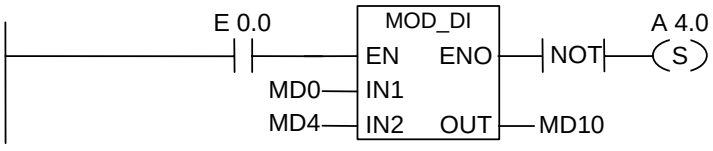
Beschreibung

MOD_DI (Divisionsrest erzeugen) dividiert IN1 durch IN2, wenn der Signalzustand "1" am Freigabeeingang EN anliegt. Der Divisionsrest kann an Ausgang OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), sind die OV- und OS-Bits = 1 und ENO = 0, so daß andere Operationen, die über ENO verknüpft sind (Kaskadenschaltung), nach dieser arithmetischen Operation nicht ausgeführt werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

Beispiel



Die Box DIV_DI wird aktiviert, wenn E 0.0 = 1 ist. Der Divisionsrest der Division MD0 durch MD4 wird von MD10 ausgegeben. Liegt der Divisionsrest außerhalb des zulässigen Bereichs für Ganzzahlen (32 Bit), oder ist der Signalzustand von E 0.0 = 0, dann wird Ausgang A 4.0 gesetzt.

8 Gleitpunkt-Funktionen

8.1 Gleitpunkt-Funktionen Übersicht

Beschreibung

Die Gleitpunktzahlen gehören zum Datentyp REAL. Mit den Gleitpunkt-Funktionen können Sie folgende arithmetische Operationen mit **zwei Gleitpunktzahlen** (32 Bit, IEEE-FP) ausführen:

- ADD_R Addieren
- SUB_R Subtrahieren
- MUL_R Multiplizieren
- DIV_R Dividieren

Folgende Funktionen können Sie mit **einer Gleitpunktzahl** (32 Bit, IEEE-FP) ausführen:

- Bilden des Absolutwertes (ABS)
- Bilden des Quadrats (SQR) bzw. der Quadratwurzel (SQRT)
- Bilden des natürlichen Logarithmus (LN)
- Bilden des Exponentialwertes (EXP) auf der Basis e (= 2,71828)
- Bilden von trigonometrischen Funktionen von einem Winkel, der als Gleitpunktzahl dargestellt ist
  - Sinus (SIN) und Arcussinus (ASIN)
  - Cosinus (COS) und Arcuscosinus (ACOS)
  - Tangens (TAN) und Arcustanges (ATAN)

8.2 Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen

Beschreibung

Die Gleitpunkt-Funktionen beeinflussen die Bits A1, A0, OV und OS im Statuswort.

Die folgenden Tabellen zeigen den Signalzustand der Bits im Statuswort für die Ergebnisse von Operationen mit Gleitpunktzahlen (32 Bit).

Gültiger Bereich	A1	A0	OV	OS
+0, -0 (Null)	0	0	0	*
-3,402823E+38 < Ergebnis < -1,175494E-38 (negative Zahl)	0	1	0	*
+1,175494E-38 < Ergebnis < 3,402824E+38 (positive Zahl)	1	0	0	*

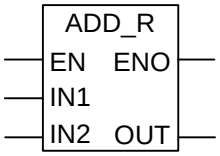
* Das OS-Bit wird vom Ergebnis der Operation nicht beeinflußt.

Ungültiger Bereich	A1	A0	OV	OS
Unterschreitung -1,175494E-38 < Ergebnis < - 1,401298E-45 (negative Zahl)	0	0	1	1
Unterschreitung +1,401298E-45 < Ergebnis < +1,175494E-38 (positive Zahl)	0	0	1	1
Überlauf Ergebnis < -3,402823E+38 (negative Zahl)	0	1	1	1
Überlauf Ergebnis > 3,402823E+38 (positive Zahl)	1	0	1	1
keine gültige Gleitpunktzahl oder unzulässige Operation (Eingangswert außerhalb des gültigen Wertebereichs)	1	1	1	1

8.3 Grundoperationen

8.3.1 ADD_R Gleitpunktzahlen addieren

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	REAL	E, A, M, L, D oder Konstante	Erster Wert der Addition
IN2	REAL	E, A, M, L, D oder Konstante	Zweiter Wert der Addition
OUT	REAL	E, A, M, L, D	Ergebnis der Addition

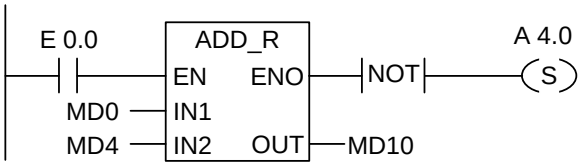
Beschreibung

ADD_R (Gleitpunktzahlen addieren) addiert IN1 und IN2, wenn der Signalzustand "1" am Freigabeeingang (EN) anliegt. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen (Überlauf bzw. Unterschreitung), sind das OV-Bit und das OS-Bit = 1 und ENO = 0, so daß andere Operationen nach dieser arithmetischen Operation, die über ENO verknüpft sind (Kaskadenschaltung), nicht ausgeführt werden. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

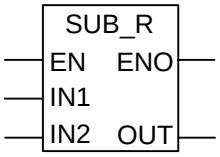
Beispiel



Die Box ADD_R wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Addition MD0 + MD4 wird an MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen oder wird diese Anweisung nicht bearbeitet (E 0.0 = 0), dann wird Ausgang A 4.0 gesetzt.

8.3.2 SUB_R Gleitpunktzahlen subtrahieren

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	REAL	E, A, M, L, D oder Konstante	Erster Wert der Subtraktion
IN2	REAL	E, A, M, L, D oder Konstante	Zweiter Wert der Subtraktion
OUT	REAL	E, A, M, L, D	Ergebnis der Subtraktion

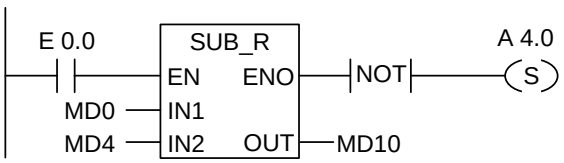
Beschreibung

SUB_R (Gleitpunktzahlen subtrahieren) subtrahiert IN2 von IN1, wenn der Signalzustand "1" am Freigabeeingang (EN) anliegt. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen (Überlauf bzw. Unterschreitung), sind das OV-Bit und das OS-Bit = 1 und ENO = 0, so daß andere Operationen nach dieser arithmetischen Operation, die über ENO verknüpft sind (Kaskadenschaltung), nicht ausgeführt werden. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

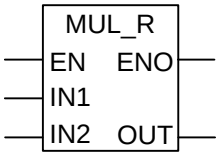
Beispiel



Die Box SUB_R wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Subtraktion MD0 - MD4 wird an MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen oder wird diese Anweisung nicht bearbeitet (E 0.0 = 0), dann wird Ausgang A 4.0 gesetzt.

8.3.3 MUL_R Gleitpunktzahlen multiplizieren

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	REAL	E, A, M, L, D oder Konstante	Erster Wert der Multiplikation
IN2	REAL	E, A, M, L, D oder Konstante	Zweiter Wert der Multiplikation
OUT	REAL	E, A, M, L, D	Ergebnis der Multiplikation

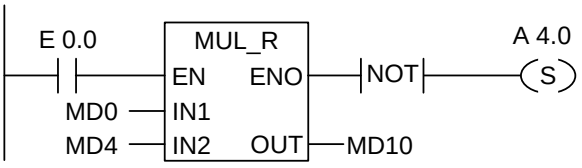
Beschreibung

MUL_R (Gleitpunktzahlen multiplizieren) multipliziert IN1 mit IN2, wenn der Signalzustand "1" am Freigabeeingang (EN) anliegt. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen (Überlauf bzw. Unterschreitung), sind das OV-Bit und OS-Bit = 1 und ENO = 0, so daß andere Operationen nach dieser arithmetischen Operation, die über ENO verknüpft sind (Kaskadenschaltung), nicht ausgeführt werden. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

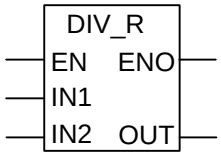
Beispiel



Die Box MUL_R wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Multiplikation MD0 X MD4 wird an MD0 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen oder wird diese Anweisung nicht bearbeitet (E 0.0 = 0), dann wird Ausgang A 4.0 gesetzt.

8.3.4 DIV_R Gleitpunktzahlen dividieren

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	REAL	E, A, M, L, D oder Konstante	Dividend
IN2	REAL	E, A, M, L, D oder Konstante	Divisor
OUT	REAL	E, A, M, L, D	Ergebnis der Division

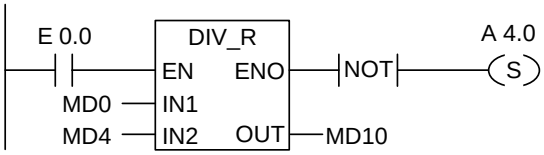
Beschreibung

DIV_R (Gleitpunktzahlen dividieren) dividiert IN1 durch IN2, wenn der Signalzustand "1" am Freigabeeingang (EN) anliegt. Das Ergebnis kann an OUT abgefragt werden. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen (Überlauf bzw. Unterschreitung), sind das OV-Bit und OS-Bit = 1 und ENO = 0, so daß andere Operationen nach dieser arithmetischen Operation, die über ENO verknüpft sind (Kaskadenschaltung), nicht ausgeführt werden. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

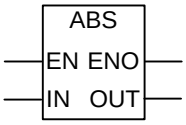
Beispiel



Die Box DIV_R wird aktiviert, wenn E 0.0 = 1 ist. Das Ergebnis der Division MD0 durch MD4 wird an MD10 ausgegeben. Liegt das Ergebnis außerhalb des zulässigen Bereichs für Gleitpunktzahlen oder wird diese Anweisung nicht bearbeitet (E 0.0 = 0), dann wird Ausgang A 4.0 gesetzt.

8.3.5 ABS Bilden des Absolutwertes einer Gleitpunktzahl

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Absolutwert der Gleitpunktzahl

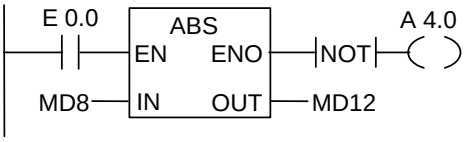
Beschreibung

ABS (Bilden des Absolutwertes einer Gleitpunktzahl) bildet den Absolutwert einer Gleitpunktzahl.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

Beispiel



Ist E 0.0 = 1, dann wird der Absolutwert von MD8 an MD12 ausgegeben.

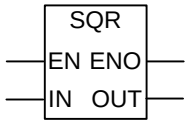
MD8 = + 6,234 -> MD12 = 6,234

Ausgang A 4.0 ist "1", wenn die Umwandlung nicht ausgeführt wird (ENO = EN = 0).

8.4 Erweiterte Operationen

8.4.1 SQR Bilden des Quadrats

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Quadrat der Gleitpunktzahl

Beschreibung

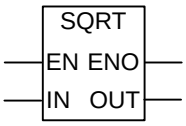
SQR (Bilden des Quadrats einer Gleitpunktzahl) bildet das Quadrat einer Gleitpunktzahl. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.2 SQRT Bilden der Quadratwurzel

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Quadratwurzel der Gleitpunktzahl

Beschreibung

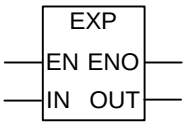
SQRT (Bilden der Quadratwurzel einer Gleitpunktzahl) bildet die Quadratwurzel einer Gleitpunktzahl. Diese Operation gibt ein positives Ergebnis aus, wenn der Operand größer als "0" ist. Einzige Ausnahme: Die Quadratwurzel von -0 ist -0. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.3 EXP Bilden des Exponentialwerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Exponentialwert der Gleitpunktzahl

Beschreibung

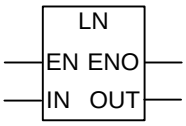
EXP (Bilden des Exponentialwerts einer Gleitpunktzahl) bildet den Exponentialwert einer Gleitpunktzahl auf der Basis e (= 2,71828...). Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.4 LN Bilden des natürlichen Logarithmus

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Natürlicher Logarithmus der Gleitpunktzahl

Beschreibung

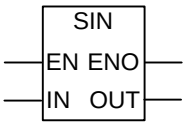
LN (Bilden des natürlichen Logarithmus einer Gleitpunktzahl) bildet den natürlichen Logarithmus einer Gleitpunktzahl. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.5 SIN Bilden des Sinuswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Sinus der Gleitpunktzahl

Beschreibung

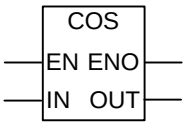
SIN (Bilden des Sinuswerts einer Gleitpunktzahl) bildet den Sinuswert einer Gleitpunktzahl. Die Gleitpunktzahl stellt dabei einen Winkel im Bogenmaß dar. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.6 COS Bildes des Cosinuswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Cosinus der Gleitpunktzahl

Beschreibung

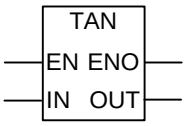
COS (Bilden des Cosinuswerts einer Gleitpunktzahl) bildet den Cosinuswert einer Gleitpunktzahl. Die Gleitpunktzahl stellt dabei einen Winkel im Bogenmaß dar. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.7 TAN Bilden des Tangenswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Tangens der Gleitpunktzahl

Beschreibung

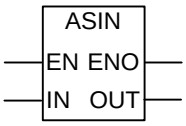
TAN (Bilden des Tangenswerts einer Gleitpunktzahl) bildet den Tangenswert einer Gleitpunktzahl. Die Gleitpunktzahl stellt dabei einen Winkel im Bogenmaß dar. Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.8 ASIN Bilden des Arcussinuswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Arcussinus der Gleitpunktzahl

Beschreibung

ASIN (Bilden des Arcussinuswerts einer Gleitpunktzahl) bildet den Arcussinuswert einer Gleitpunktzahl, deren Definitionsbereich -1 <= Eingangswert <= 1 ist. Das Ergebnis stellt dabei einen Winkel im Bogenmaß dar. Der Wert liegt im folgenden Bereich:

$$-\pi/2 \leq \text{Ausgangswert} \leq +\pi/2$$

$$\pi = 3{,}1415...$$

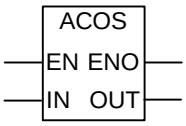
Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.9 ACOS Bilden des Arcuscosinuswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Arcuscosinus der Gleitpunktzahl

Beschreibung

ACOS (Bilden des Arcuscosinuswerts einer Gleitpunktzahl) bildet den Arcuscosinuswert einer Gleitpunktzahl, deren Definitionsbereich -1 <= Eingangswert <= 1 ist. Das Ergebnis stellt dabei einen Winkel im Bogenmaß dar. Der Wert liegt im folgenden Bereich:

$$-\pi/2 \leq \text{Ausgangswert} \leq +\pi/2$$

$$\pi = 3{,}1415...$$

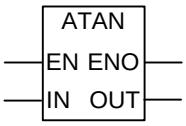
Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

8.4.10 ATAN Bilden des Arcustangenswerts

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	REAL	E, A, M, L, D oder Konstante	Eingangswert: Gleitpunktzahl
OUT	REAL	E, A, M, L, D	Ausgangswert: Arcustangens der Gleitpunktzahl

Beschreibung

ATAN (Bilden des Arcustangenswerts einer Gleitpunktzahl) bildet den Arcustangenswert einer Gleitpunktzahl. Das Ergebnis stellt dabei einen Winkel im Bogenmaß dar. Der Wert liegt im folgenden Bereich:

$$-\pi/2 \leq \text{Ausgangswert} \leq +\pi/2$$

$$\pi = 3{,}1415...$$

Siehe auch Auswerten der Bits im Statuswort bei Gleitpunkt-Funktionen.

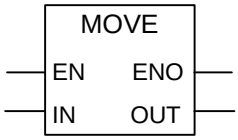
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	X	0	X	X	1

9 Verschieben

9.1 MOVE Wert übertragen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	Alle elementaren Datentypen mit einer Länge von 8, 16 oder 32 Bit	E, A, M, L, D oder Konstante	Quellwert
OUT	Alle elementaren Datentypen mit einer Länge von 8, 16 oder 32 Bit	E, A, M, L, D	Zieladresse

Beschreibung

MOVE (Wert übertragen) wird vom Freigabeeingang EN aktiviert. Der Wert, der vom Eingang IN angegeben wird, wird an die Adresse, die vom Ausgang OUT angegeben wird, kopiert. ENO hat den gleichen Signalzustand wie EN. Die Operation **MOVE** kann nur die Datenobjekte von der Länge BYTE, WORD oder DWORD kopieren. Anwenderdefinierte Datentypen wie Felder oder Strukturen müssen mit dem SFC 20 "BLKMOV" kopiert werden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	-	-	-	-	0	1	1	1

Abhängigkeit vom MCR (Master Control Relay)

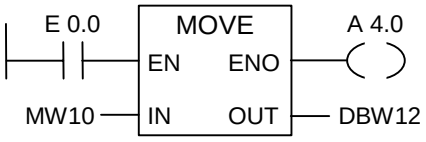
Die MCR-Abhängigkeit wird nur aktiviert, wenn die Box MOVE innerhalb eines aktiven MCR-Bereichs angeordnet wird. Innerhalb eines aktivierten MCR-Bereichs werden die adressierten Daten wie oben beschrieben kopiert, sofern das MCR eingeschaltet ist und Signalfluß am Freigabeeingang vorhanden ist. Ist das MCR ausgeschaltet und wird eine Operation **MOVE** ausgeführt, so wird unabhängig vom aktuellen Zustand von IN der Wert "0" an die von OUT angegebene Adresse geschrieben.

Hinweis

Bei der Übertragung eines Wertes in einen Datentyp anderer Länge werden höherwertige Bytes bei Bedarf abgeschnitten oder mit Nullen aufgefüllt. Beispiele:

Doppelwort	**1111 1111**	**0000 1111**	**1111 0000**	**0101 0101**
Übertragung	**Ergebnis**			
in ein Doppelwort:	1111 1111	0000 1111	1111 0000	0101 0101
in ein Byte:				0101 0101
in ein Wort:			1111 0000	0101 0101
Byte				**1111 0000**
Übertragung	**Ergebnis**			
in ein Byte:				1111 0000
in ein Wort:			0000 0000	1111 0000
in ein Doppelwort:	0000 0000	0000 0000	0000 0000	1111 0000

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist. Der Inhalt von MW10 wird dann in das Datenwort 12 des aktuell geöffneten Datenbausteins kopiert.

A 4.0 ist "1", wenn die Operation ausgeführt wird.

Befindet sich der Strompfad aus dem Beispiel in einem aktiven MCR-Bereich:

- Ist das MCR eingeschaltet, dann werden die Daten wie oben beschrieben von MW10 in DBW12 kopiert.
- Ist das MCR ausgeschaltet, dann wird der Wert "0" in DBW12 geschrieben.

10 Programmsteuerung

10.1 Programmsteuerungsoperationen Übersicht

Beschreibung

Folgende Operationen stehen Ihnen zur Programmsteuerung zur Verfügung:

- ---(Call) FC/SFC aufrufen ohne Parameter
- CALL_FB FB als Box aufrufen
- CALL_FC FC als Box aufrufen
- CALL_SFB System-FB als Box aufrufen
- CALL_SFC System-FC als Box aufrufen
- Multiinstanzen aufrufen
- Baustein aus einer Bibliothek aufrufen

- Wichtige Hinweise zur MCR-Funktionalität
- ---(MCR<) Master Control Relay einschalten
- ---(MCR>) Master Control Relay ausschalten
- ---(MCRA) Master Control Relay Anfang
- ---(MCRD) Master Control Relay Ende
- RET Springe zurück

10.2 ---(Call) FC/SFC aufrufen ohne Parameter

Symbol

<FC/SFC-Nr:>

---(CALL)

Parameter	Datentyp	Speicherbereich	Beschreibung
<FC/SFC-Nr.>	BLOCK_FC	-	Nummer der FC/SFC. Welche SFCs zur Verfügung stehen, hängt von Ihrer CPU ab.

Beschreibung

---(Call) (FC/SFC aufrufen ohne Parameter) ruft eine Funktion (FC) oder eine Systemfunktion (SFC) auf, die keine Parameter hat. Ein Aufruf wird nur ausgeführt, wenn das VKE an der CALL-Spule "1" ist. Die Operation **---(Call)** arbeitet folgendermaßen:

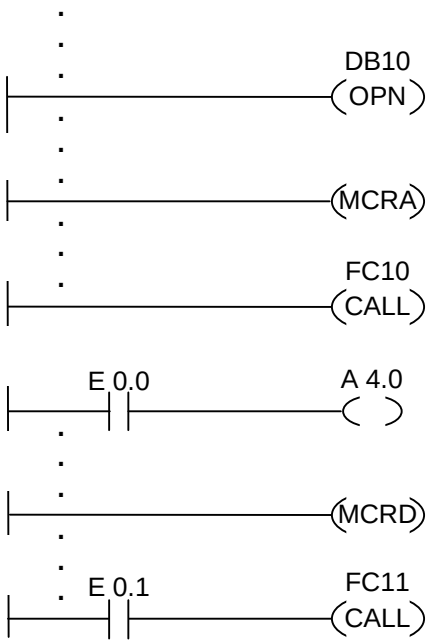
- Sie speichert die Rücksprungadresse des aufrufenden Bausteins.
- Sie speichert die beiden Datenbaustein-Register (Datenbaustein und Instanz-Datenbaustein).
- Sie aktualisiert den Lokaldatenbereich für die aufgerufene FC oder SFC.
- Sie schiebt das MA-Bit (aktives MCR-Bit) in den Baustein-Stack (B-Stack).

Anschließend wird die Programmbearbeitung in der aufgerufenen Funktion oder Systemfunktion fortgesetzt.

Statuswort

		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
Bedingt:	schreibt	-	-	-	-	0	0	1	1	0
Absolut:	schreibt	-	-	-	-	0	0	1	-	0

Beispiel



Bei den oben dargestellten Strompfaden des Kontaktplans handelt es sich um Programmteile eines vom Anwender geschriebenen Funktionsbausteins. In diesem Funktionsbaustein wird DB 10 geöffnet und das MCR aktiviert. Wird der absolute Aufruf von FC 10 ausgeführt, geschieht folgendes:

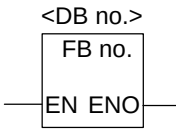
Die Rücksprungadresse des aufrufenden Funktionsbausteins und die Auswahldaten für den DB 10 und für den Instanz-Datenbaustein des aufrufenden Funktionsbausteins werden gespeichert. Das MA-Bit, das von der Operation MCRA auf "1" gesetzt wurde, wird in den B-Stack geschoben und dann für den aufgerufenen Baustein (FC 10) auf "0" gesetzt. Die Programmbearbeitung wird in FC 10 fortgesetzt. Benötigt FC 10 das MCR, muß das MCR in FC 10 wieder aktiviert werden. Ist die Bearbeitung von FC 10 beendet, geht die Programmbearbeitung zurück zum aufrufenden FB. Das MA-Bit wird wiederhergestellt. DB 10 und der Instanz-Datenbaustein des vom Anwender geschriebenen Funktionsbausteins werden wieder die aktuellen DBs. Das Programm wird im nächsten Strompfad fortgesetzt, in dem dem Ausgang A 4.0 der Zustand von E 0.0 zugeordnet wird. Beim Aufruf von FC 11 handelt es sich um einen bedingten Aufruf. Dieser Aufruf wird nur ausgeführt, wenn E 0.1 = 1 ist. Wird der Aufruf ausgeführt, wird die Programmsteuerung wie für FC 10 beschrieben an FC 11 übergeben und kehrt nach der Bearbeitung von FC 11 zurück.

Hinweis

Nach dem Rücksprung in den aufrufenden Baustein ist nicht immer sichergestellt, daß der zuvor geöffnete DB wieder geöffnet ist. Beachten Sie bitte den Hinweis in der Liesmich-Datei.

10.3 CALL_FB FB als Box aufrufen

Symbol



Das Symbol ist von dem Funktionsbaustein abhängig (je nachdem, ob bzw. wie viele Parameter vorhanden sind). EN, ENO und der Name bzw. die Nummer des FB müssen vorhanden sein.

Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
FB no. DB no.	BLOCK_FB BLOCK_DB	- -	Nummer des FB/DB, Bereich ist von der CPU abhängig

Beschreibung

CALL_FB (FB als Box aufrufen) wird ausgeführt, wenn EN = 1 ist. Die Operation arbeitet folgendermaßen:

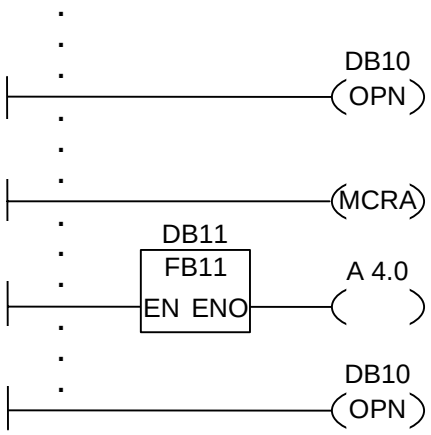
- Sie speichert die Rücksprungadresse des aufrufenden Bausteins.
- Sie speichert die Auswahldaten für die beiden aktuellen Datenbausteine (DB und Instanz-DB).
- Sie aktualisiert den Lokaldatenbereich für den aufgerufenen Funktionsbaustein.
- Sie schiebt das MA-Bit (aktives MCR-Bit) in den Baustein-Stack (B-Stack).

Anschließend wird die Programmbearbeitung in dem aufgerufenen Funktionsbaustein fortgesetzt. Zur Ermittlung des ENO wird das BIE-Bit abgefragt, diesem muß vom Anwender im aufgerufenen Baustein mit ---(SAVE) der gewünschte Zustand (Fehlerauswertung) zugewiesen werden.

Statuswort

		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
Bedingt:	schreibt	X	-	-	-	0	0	X	X	X
Absolut:	schreibt	-	-	-	-	0	0	X	X	X

Beispiel



Bei den oben dargestellten Strompfaden eines Kontaktplans handelt es sich um Programmteile eines vom Anwender geschriebenen Funktionsbausteins. DB 10 wird in diesem Baustein geöffnet und das MCR aktiviert. Wenn der absolute Aufruf von FB 11 ausgeführt wird, geschieht folgendes:

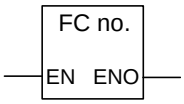
Die Rücksprungadresse des aufrufenden Funktionsbausteins und die Auswahldaten für DB 10 und den Instanz-Datenbaustein des aufrufenden Funktionsbausteins werden gespeichert. Das MA-Bit, das von der Funktion MCRA auf "1" gesetzt wurde, wird in den B-Stack geschoben und dann für den aufgerufenen Funktionsbaustein FB 11 auf "0" gesetzt. Die Programmbearbeitung wird in FB 11 fortgesetzt. Benötigt FB 11 das MCR, muß das MCR im Funktionsbaustein wieder aktiviert werden. Der Zustand des VKE muß durch die Operation ---(SAVE) im BIE-Bit gespeichert werden, um eine Fehlerauswertung im aufrufenden FB vornehmen zu können. Ist die Bearbeitung des FB 11 beendet, geht die Programmbearbeitung zurück zum aufrufenden Funktionsbaustein. Das MA-Bit wird wiederhergestellt und der Instanz-Datenbaustein des vom Anwender geschriebenen Funktionsbausteins wird wieder zum geöffneten DB. Wird der FB 11 korrekt bearbeitet, ist ENO = 1 und somit A 4.0 = 1.

Hinweis

Bei FB/SFB-Aufrufen geht die Nummer des zuvor geöffneten Datenbausteins verloren. Der benötigte DB muß erneut geöffnet werden.

10.4 CALL_FC FC als Box aufrufen

Symbol



Das Symbol ist von der Funktion abhängig (je nachdem, ob bzw. wie viele Parameter vorhanden sind). EN, ENO und der Name bzw. die Nummer der FC müssen vorhanden sein.

Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
FC no.	BLOCK_FC	-	Nummer der FC, Bereich ist von der CPU abhängig

Beschreibung

CALL_FC (FC als Box aufrufen) ruft eine Funktion (FC) auf. Der Aufruf wird ausgeführt, wenn EN = 1 ist. Die Operation arbeitet folgendermaßen:

- Sie speichert die Rücksprungadresse des aufrufenden Bausteins.
- Sie aktualisiert den Lokaldatenbereich für die aufgerufene Funktion.
- Sie schiebt das MA-Bit (aktives MCR-Bit) in den Baustein-Stack (B-Stack).

Anschließend wird die Programmbearbeitung in der aufgerufenen Funktion fortgesetzt.

Zur Ermittlung des ENO wird das BIE-Bit abgefragt, diesem muß vom Anwender im aufgerufenen Baustein mit ---(SAVE) der gewünschte Zustand (Fehlerauswertung) zugewiesen werden.

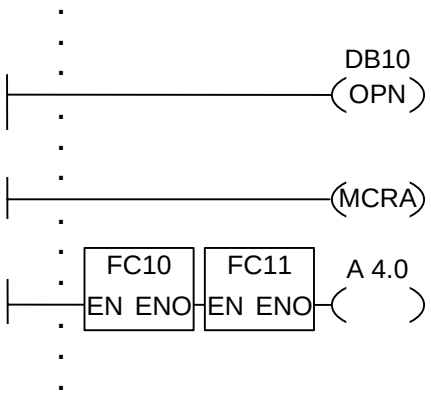
Wenn Sie eine FC aufrufen und die Variablendeklarationstabelle des aufgerufenen Bausteins über Deklarationen vom Typ IN, OUT und IN_OUT verfügt, werden diese Variablen im Programm des aufrufenden Bausteins als Liste der Formalparameter angezeigt.

Beim Aufruf der FCs **müssen Sie zwingend** den Formalparametern Aktualparameter an der Aufrufstelle zuordnen. Eventuelle Anfangswerte in der FC-Deklaration sind ohne Bedeutung.

Statuswort

		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
Bedingt:	schreibt	X	-	-	-	0	0	X	X	X
Absolut:	schreibt	-	-	-	-	0	0	X	X	X

Beispiel



Bei den oben dargestellten Strompfaden eines Kontaktplans handelt es sich um Programmteile eines vom Anwender geschriebenen Funktionsbausteins. DB 10 wird in diesem Baustein geöffnet und das MCR aktiviert. Wird der absolute Aufruf von FC 10 ausgeführt, geschieht folgendes:

Die Rücksprungadresse des aufrufenden Funktionsbausteins und die Auswahldaten für DB 10 und den Instanz-Datenbaustein des aufrufenden Funktionsbausteins werden gespeichert. Das MA-Bit, das von der Operation MCRA auf "1" gesetzt wurde, wird in den B-Stack geschoben und dann für den aufgerufenen Baustein FC 10 auf "0" gesetzt. Die Programmbearbeitung wird in FC 10 fortgesetzt. Benötigt FC 10 das MCR, dann muß das MCR in FC 10 wieder aktiviert werden. Der Zustand des VKE muß durch die Operation ---(SAVE) im BIE-Bit gespeichert werden, um eine Fehlerauswertung im aufrufenden FB vornehmen zu können. Ist die Bearbeitung von FC 10 beendet, geht die Programmbearbeitung zurück zum aufrufenden Funktionsbaustein. Das MA-Bit wird wiederhergestellt. Nach Bearbeitung der FC 10 wird in Abhängigkeit vom ENO das Programm im aufrufenden FB fortgesetzt:

ENO = 1 FC 11 wird bearbeitet

ENO = 0 Bearbeitung beginnt im nächsten Netzwerk

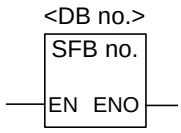
Wird auch FC 11 korrekt bearbeitet, ist ENO = 1 und somit A **4.0 = 1.**

Hinweis

Nach dem Rücksprung in den aufrufenden Baustein ist nicht immer sichergestellt, daß der zuvor geöffnete DB wieder geöffnet ist. Beachten Sie bitte den Hinweis in der Liesmich-Datei.

10.5 CALL_SFB System-FB als Box aufrufen

Symbol



Das Symbol ist von dem Systemfunktionsbaustein abhängig (je nachdem, ob bzw. wie viele Parameter vorhanden sind). EN, ENO und der Name bzw. die Nummer des SFB müssen vorhanden sein.

Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
SFB no. DB no.	BLOCK_SFB BLOCK_DB	- -	Nummer des SFB/DB, Bereich ist von der CPU abhängig

Beschreibung

CALL_SFB (SFB als Box aufrufen) wird ausgeführt, wenn EN = 1 ist. Die Operation arbeitet folgendermaßen:

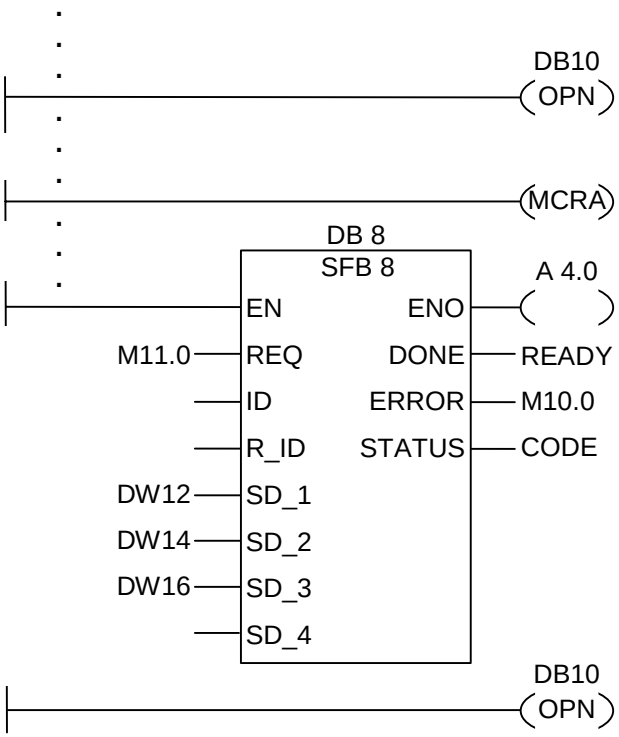
- Sie speichert die Rücksprungadresse des aufrufenden Bausteins.
- Sie speichert die Auswahldaten für die beiden aktuellen Datenbausteine (DB und Instanz-DB).
- Sie aktualisiert den Lokaldatenbereich für den aufgerufenen Systemfunktionsbaustein.
- Sie schiebt das MA-Bit (aktives MCR-Bit) in den Baustein-Stack (B-Stack).

Anschließend wird die Programmbearbeitung in dem aufgerufenen Systemfunktionsbaustein fortgesetzt. ENO ist "1", wenn der Systemfunktionsbaustein aufgerufen wurde (EN = 1) und keine Fehler aufgetreten sind.

Statuswort

		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
Bedingt:	schreibt	X	-	-	-	0	0	X	X	X
Absolut:	schreibt	-	-	-	-	0	0	X	X	X

Beispiel



Bei den oben dargestellten Strompfaden eines Kontaktplans handelt es sich um Programmteile eines vom Anwender geschriebenen Funktionsbausteins. DB 10 wird in diesem Baustein geöffnet und das MCR aktiviert. Wenn der absolute Aufruf von SFB 8 ausgeführt wird, geschieht folgendes:

Die Rücksprungadresse des aufrufenden Funktionsbausteins und die Auswahldaten für DB 10 und den Instanz-Datenbaustein des aufrufenden Funktionsbausteins werden gespeichert. Das MA-Bit, das von der Funktion MCRA auf "1" gesetzt wurde, wird in den B-Stack geschoben und dann für den aufgerufenen Systemfunktionsbaustein SFB 8 auf "0" gesetzt. Die Programmbearbeitung wird in SFB 8 fortgesetzt. Ist die Bearbeitung von SFB 8 beendet, geht die Programmbearbeitung zurück zum aufrufenden Funktionsbaustein. Das MA-Bit wird wiederhergestellt, und der Instanz-Datenbaustein des vom Anwender geschriebenen Funktionsbausteins wird wieder zum aktuellen Instanz-DB. Wird der SFB 8 korrekt bearbeitet, ist ENO = 1 und somit A4.0 = 1.

Hinweis

Bei FB/SFB-Aufrufen geht die Nummer des zuvor geöffneten Datenbausteins verloren. Der benötigte DB muß erneut geöffnet werden

10.6 CALL_SFC System-FC als Box aufrufen

Symbol

Das Symbol ist von der Systemfunktion abhängig (je nachdem, ob bzw. wie viele Parameter vorhanden sind). EN, ENO und der Name bzw. die Nummer der SFC müssen vorhanden sein.

Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
SFC no.	BLOCK_SFC	-	Nummer der SFC, Bereich ist von der CPU abhängig

Beschreibung

CALL_SFC (System-FC als Box aufrufen) ruft eine Systemfunktion auf. Der Aufruf wird ausgeführt, wenn EN = 1 ist. Die Operation arbeitet folgendermaßen:

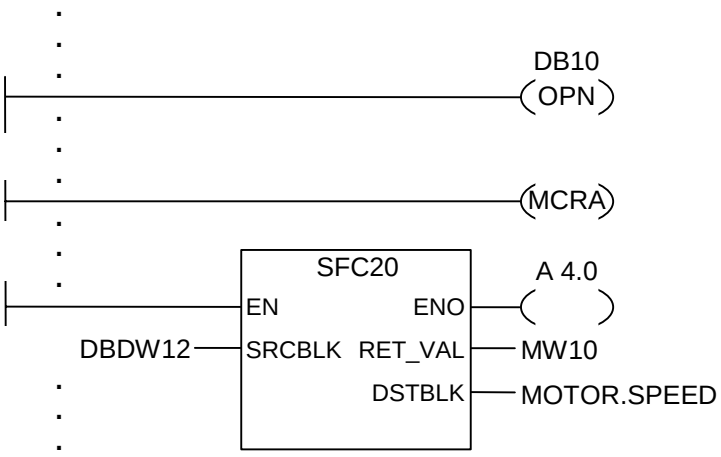
- Sie speichert die Rücksprungadresse des aufrufenden Bausteins.
- Sie aktualisiert den Lokaldatenbereich für die aufgerufene Funktion.
- Sie schiebt das MA-Bit (aktives MCR-Bit) in den Baustein-Stack (B-Stack).

Anschließend wird die Programmbearbeitung in der aufgerufenen Systemfunktion fortgesetzt. ENO ist "1", wenn die Funktion aufgerufen wurde (EN = 1) und keine Fehler aufgetreten sind.

Statuswort

		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
Bedingt:	schreibt	X	-	-	-	0	0	X	X	X
Absolut:	schreibt	-	-	-	-	0	0	X	X	X

Beispiel



Bei den oben dargestellten Strompfaden eines Kontaktplans handelt es sich um Programmteile eines vom Anwender geschriebenen Funktionsbausteins. DB 10 wird in diesem Baustein geöffnet und das MCR aktiviert. Wird der absolute Aufruf von SFC 20 ausgeführt, geschieht folgendes:

Die Rücksprungadresse des aufrufenden Funktionsbausteins und die Auswahldaten für DB 10 und den Instanz-Datenbaustein des aufrufenden Funktionsbausteins werden gespeichert. Das MA-Bit, das von der Funktion MCRA auf "1" gesetzt wurde, wird in den B-Stack geschoben und dann für den aufgerufenen Baustein SFC 20 auf "0" gesetzt. Die Programmbearbeitung wird in SFC 20 fortgesetzt. Wenn die Bearbeitung der SFC 20 beendet ist, geht die Programmbearbeitung zurück zum aufrufenden Funktionsbaustein. Das MA-Bit wird wiederhergestellt.

Nach Bearbeitung der SFC 20 wird in Abhängigkeit von ENO das Programm im aufrufenden FB fortgesetzt:

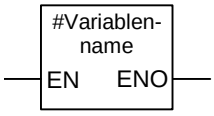
ENO = 1 A 4.0 = 1
ENO = 0 A 4.0 = 0

Hinweis

Nach dem Rücksprung in den aufrufenden Baustein ist nicht immer sichergestellt, daß der zuvor geöffnete DB wieder geöffnet ist. Beachten Sie bitte den Hinweis in der Liesmich-Datei.

10.7 Multiinstanz aufrufen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
# Variablenname	FB, SFB	-	Name der Multiinstanz

Beschreibung

Eine Multiinstanz entsteht durch die Deklaration einer statischen Variable vom Datentyp eines Funktionsbausteins. Nur bereits deklarierte Multiinstanzen werden im Programmelementekatalog aufgeführt. Das Symbol einer Multiinstanz verändert sich, je nachdem, ob und wie viele Parameter vorhanden sind. EN, ENO und der Variablenname sind immer vorhanden.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	0	0	X	X	X

10.8 Baustein aus einer Bibliothek aufrufen

Die im SIMATIC Manager bekannten Bibliotheken werden Ihnen zur Auswahl angeboten.

Aus diesen Bibliotheken können Sie Bausteine auswählen,

- die im Betriebssystem Ihrer CPU integriert sind (Bibliothek "Standard Library" für STEP 7-Projekte der Version 3 und "stdlibs (V2)" für STEP 7-Projekte der Version 2),
- die Sie selbst in Bibliotheken abgelegt haben, weil Sie sie mehrfach verwenden wollen.

10.9 Wichtige Hinweise zur MCR-Funktionalität

Vorsicht bei Bausteinen, in denen mit MCRA das Master Control Relay aktiviert wurde:

- Wenn das MCR abgeschaltet ist, wird in Programmabschnitten zwischen ---(MCR<) und ---(MCR>) durch alle Zuweisungen der Wert 0 geschrieben! Das betrifft dann natürlich auch **alle** Boxen, die eine Zuweisung enthalten, einschließlich der Parameterübergabe an Bausteine!
- Das MCR ist genau dann abgeschaltet, wenn vor einem **MCR<**-Befehl das VKE = 0 war.

Gefahr: STOP der AS oder undefiniertes Laufzeitverhalten !

Der Compiler greift für Adreßberechnungen auch schreibend auf Lokaldaten hinter den in VAR_TEMP definierten temporären Variablen zu. Daher setzen folgende Befehlssequenzen die AS in STOP oder führen zu undefiniertem Laufzeitverhalten:

Formalparameterzugriffe

- Zugriffe auf Komponenten komplexer FC-Parameter vom Typ STRUCT, UDT, ARRAY, STRING
- Zugriffe auf Komponenten komplexer FB-Parameter vom Typ STRUCT, UDT, ARRAY, STRING aus dem Bereich IN_OUT in einem multiinstanzfähigen Baustein (Bausteinversion 2).
- Zugriffe auf Parameter eines multiinstanzfähigen FB (Bausteinversion 2), wenn ihre Adresse größer als 8180.0 ist.
- Zugriff im multiinstanzfähigen FB (Bausteinversion 2) auf einen Parameter vom Typ BLOCK_DB schlägt den DB 0 auf. Nachfolgende Datenzugriffe bringen die CPU in STOP. Bei TIMER, COUNTER, BLOCK_FC, BLOCK_FB wird auch immer T 0, Z 0, FC 0 bzw. FB 0 verwendet.

Parameterübergabe

- Calls, bei denen Parameter übergeben werden.

KOP/FUP

- T-Abzweige und Konnektoren in KOP oder FUP starten mit VKE = 0.

Abhilfe

Lösen Sie die genannten Befehle aus der MCR-Abhängigkeit:

1. Deaktivieren Sie das MCR mit Master Control Relay Ende **vor** der betreffenden Anweisung bzw. vor dem betreffenden Netzwerk.
2. Aktivieren Sie das MCR mit Master Control Relay Anfang **nach** der betreffenden Anweisung bzw. nach dem betreffenden Netzwerk.

10.10 ---(MCR<) Master Control Relay einschalten

Wichtige Hinweise zur MCR-Funktionalität

Symbol

---(MCR<)

Beschreibung

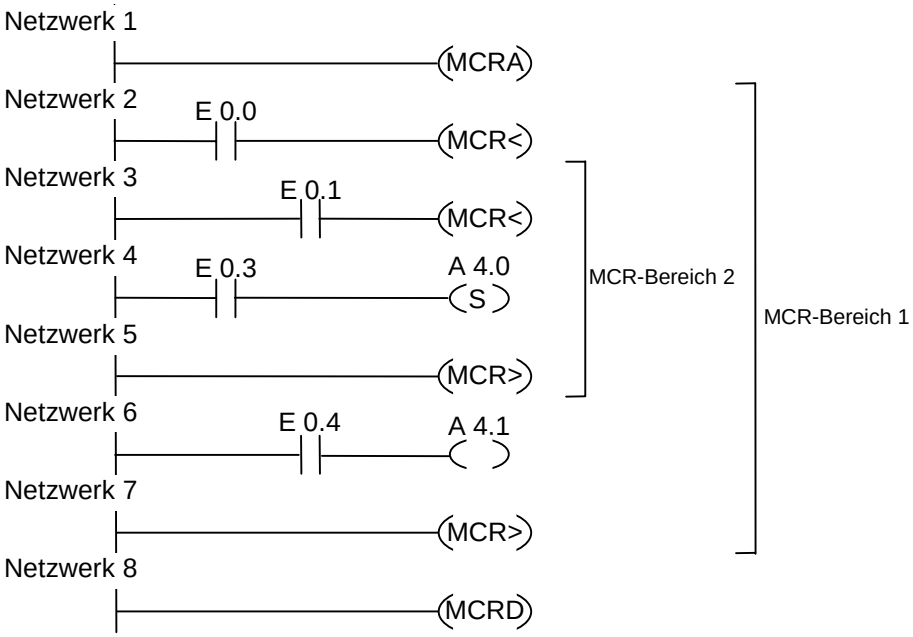
---(MCR<) (Master Control Relay einschalten) speichert das VKE im MCR-Stack und öffnet einen MCR-Bereich. Beim MCR-Klammerstack handelt es sich um einen LIFO-Stack (last in, first out), der maximal 8 Stackeinträge aufnehmen kann (8 Ebenen). Ist der Stack bereits voll, ruft die Operation **---(MCR<)** einen MCR-Stack-Fehler hervor (MCRF). Die folgenden Elemente sind vom MCR abhängig und werden vom Signalzustand des VKE beeinflußt, der im MCR-Stack gespeichert wird, solange ein MCR-Bereich geöffnet ist:

- --(#) Konnektor
- --( ) Relaisspule, Ausgang
- --(S) Ausgang setzen
- --(R) Ausgang rücksetzen
- RS Flipflop rücksetzen setzen
- SR Flipflop setzen rücksetzen
- MOVE Wert übertragen

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	1	-	0

Beispiel



Das MCR wird vom Strompfad MCRA aktiviert. Dann können maximal acht geschachtelte MCR-Bereiche erstellt werden. Im Beispiel gibt es zwei MCR-Bereiche. Die Operationen werden folgendermaßen ausgeführt:

E 0.0 = 1 (das MCR ist EIN in Bereich 1): der Signalzustand von E 0.4 wird A 4.1 zugeordnet

E 0.0 = 0 (das MCR ist AUS in Bereich 1): A 4.1 ist "0", unabhängig vom Zustand von E 0.4

E 0.0 UND E 0.1 = 1 (das MCR ist EIN in Bereich 2): A 4.0 wird auf "1" gesetzt, wenn E 0.3 = 1

E 0.0 UND E 0.1 = 0 (das MCR ist AUS in Bereich 2): A 4.0 wird nicht verändert, unabhängig vom Zustand von E 0.3

10.11 ---(MCR>) Master Control Relay ausschalten

Wichtige Hinweise zur MCR-Funktionalität

Symbol

---(MCR>)

Beschreibung

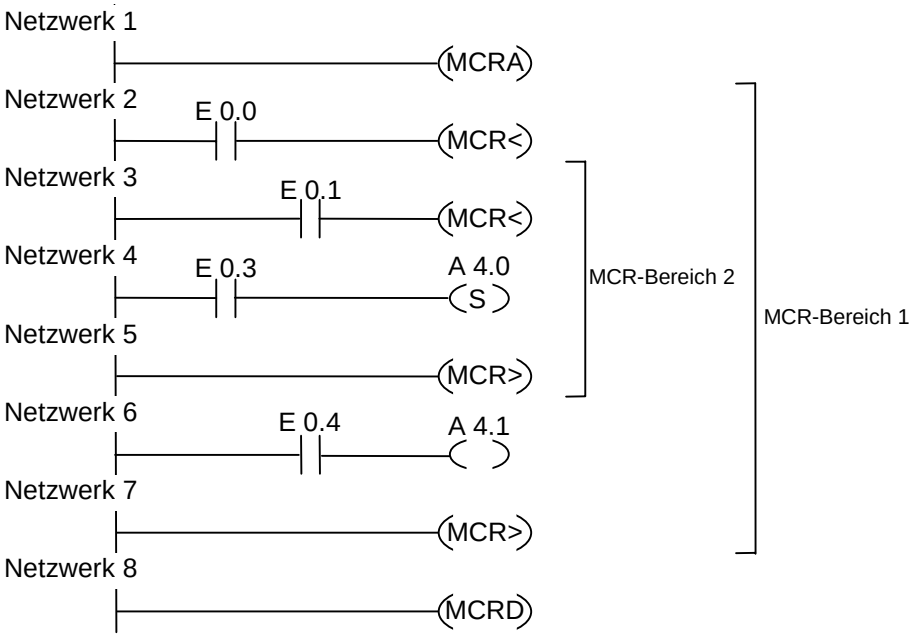
---(MCR>) (Master Control Relay ausschalten) löscht einen VKE-Eintrag aus dem MCR-Stack. Bei dem MCR-Klammerstack handelt es sich um einen LIFO-Stack (last in, first out), der maximal 8 Stackeinträge aufnehmen kann (8 Ebenen). Ist der Stack bereits leer, ruft die Operation **---(MCR>)** einen MCR-Stack-Fehler hervor (MCRF). Die folgenden Elemente sind vom MCR abhängig und werden von dem Signalzustand des VKE beeinflußt, der im MCR-Stack gespeichert wird, solange ein MCR-Bereich geöffnet ist:

- --(#) Konnektor
- --( ) Relaisspule, Ausgang
- --(S) Ausgang setzen
- --(R) Ausgang rücksetzen
- RS Flipflop rücksetzen, setzen
- SR Flipflop setzen rücksetzen
- MOVE Wert übertragen

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	1	-	0

Beispiel



Das MCR wird von der Operation ---(MCRA) aktiviert. Dann können Sie maximal acht MCR-Bereiche erstellen. In diesem Beispiel gibt es zwei MCR-Bereiche. Der erste Strompfad ---(MCR>) (MCR AUS) gehört zum zweiten Strompfad ---(MCR<) (MCR EIN). Alle Strompfade zwischen diesen beiden gehören zum MCR-Bereich 2. Die Funktionen werden folgendermaßen ausgeführt:

E 0.0 = 1: der Signalzustand von E 0.4 wird dem Ausgang A 4.1 zugeordnet

E 0.0 = 0: A 4.1 ist "0", unabhängig von dem Zustand von E 0.4

E 0.0 UND E 0.1 = 1: A 4.0 wird auf "1" gesetzt, wenn E 0.3 = 1 ist

E 0.0 UND E 0.1 = 0: A 4.0 wird nicht verändert, unabhängig vom Zustand von E 0.3

10.12 ---(MCRA) Master Control Relay Anfang

Wichtige Hinweise zur MCR-Funktionalität

Symbol

---(MCRA)

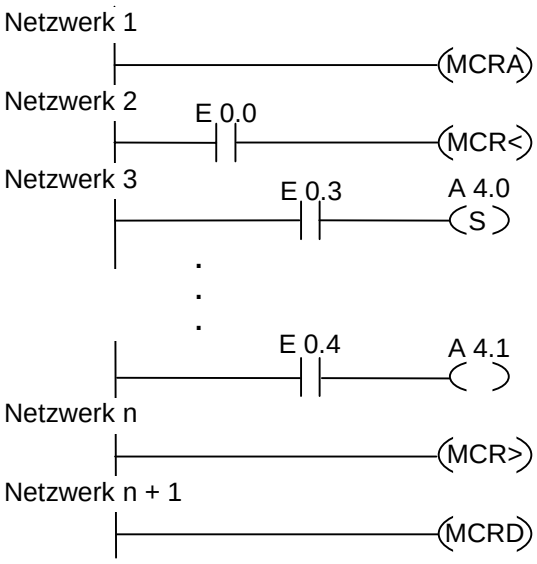
Beschreibung

---(MCRA) (Master Control Relay Anfang) aktiviert das Master Control Relay. Nach dieser Operation können Sie mit den Operationen ---(MCR<) und ---(MCR>) MCR-Bereiche programmieren.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	-	-	-	-

Beispiel



Das MCR wird vom Strompfad MCRA aktiviert. Die Strompfade zwischen den Operationen MCR< und MCR> (Ausgänge A 4.0, A 4.1) werden folgendermaßen ausgeführt:

E 0.0 = 1 (MCR ist EIN): A 4.0 wird auf "1" gesetzt, wenn E 0.3 im Zustand "1" ist bzw. wird nicht verändert, wenn E 0.3 im Zustand "0" ist. Der Zustand von E 0.4 wird dem Ausgang A 4.1 zugeordnet.

E 0.0 = 0 (MCR ist AUS): A 4.0 wird nicht verändert, unabhängig vom Zustand von E 0.3. A 4.1 ist "0", unabhängig vom Zustand von E0.4.

Im nächsten Strompfad deaktiviert die Operation ---(MCRD) das MCR. Das bedeutet, daß Sie mit dem Operationspaar ---(MCR<) und ---(MCR>) keine MCR-Bereiche mehr programmieren können.

10.13 ---(MCRD) Master Control Relay Ende

Wichtige Hinweise zur MCR-Funktionalität

Symbol

---(MCRD)

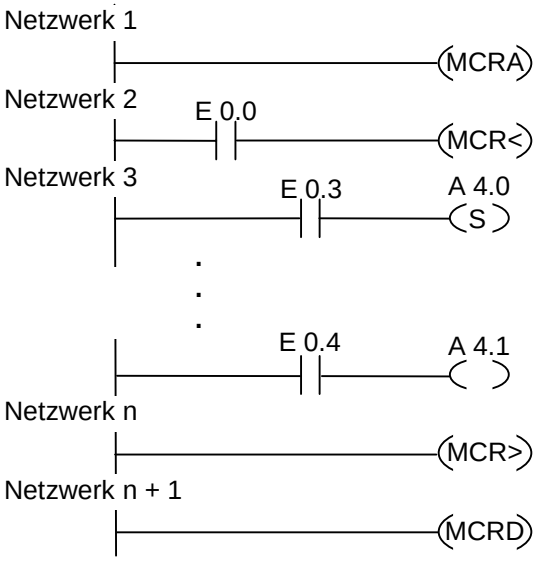
Beschreibung

---(MCRD) (Master Control Relay Ende) deaktiviert das MCR. Nach der Ausführung dieser Operation können Sie keine MCR-Bereiche programmieren.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	-	-	-	-

Beispiel



Das MCR wird vom Strompfad MCRA aktiviert. Die Strompfade zwischen den Operationen MCR< und MCR> (Ausgänge A 4.0, A 4.1) werden folgendermaßen ausgeführt:

E 0.0 = 1 (MCR ist EIN): A 4.0 wird auf "1" gesetzt, wenn E 0.3 im Zustand "1" ist bzw. wird nicht verändert, wenn E 0.3 im Zustand "0" ist. Der Zustand von E 0.4 wird dem Ausgang A 4.1 zugeordnet.

E 0.0 = 0 (MCR ist AUS): A 4.0 wird nicht verändert, unabhängig vom Zustand von E 0.3. A 4.1 ist "0", unabhängig vom Zustand von E 0.4.

Im nächsten Strompfad deaktiviert die Operation ---(MCRD) das MCR. Das bedeutet, daß Sie mit dem Operationspaar ---(MCR<) und ---(MCR>) keine MCR-Bereiche mehr programmieren können.

10.14 ---(RET) Springe zurück

Symbol

---(RET)

Beschreibung

RET (Springe zurück) dient zum bedingten Verlassen von Bausteinen. Bei diesem Ausgang ist eine Vorverknüpfung erforderlich.

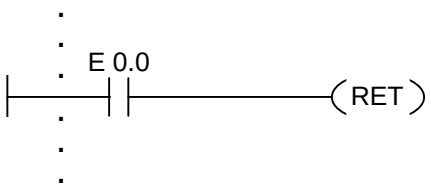
Statuswort

Bedingter Rücksprung (Rücksprung, wenn VKE = 1):

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	*	-	-	-	0	0	1	1	0

* Die Operation **RET** wird intern auf die Sequenz "SAVE; BEB;" abgebildet. Das bewirkt, daß auch das BIE-Bit beeinflußt wird.

Beispiel



Der Baustein wird verlassen, wenn E 0.0 = 1 ist.

11 Schieben/Rotieren

11.1 Schiebeoperationen

11.1.1 Schiebeoperationen Übersicht

Beschreibung

Mit den Schiebeoperationen können Sie den Inhalt von Eingang IN bitweise nach links oder rechts schieben (siehe auch CPU-Register). Ein Schieben um n Bits nach links multipliziert den Inhalt von Eingang IN mit 2 hoch n; ein Schieben um n Bits nach rechts dividiert den Inhalt von Eingang IN durch 2 hoch n. Wenn Sie also beispielsweise das binäre Äquivalent des Dezimalwerts 3 um 3 Bits nach links schieben, so ergibt sich das binäre Äquivalent des Dezimalwerts 24. Schieben Sie das binäre Äquivalent des Dezimalwerts 16 um 2 Bits nach rechts, so ergibt sich das binäre Äquivalent des Dezimalwerts 4.

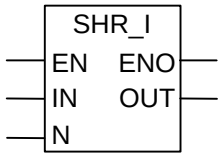
Am Eingang N können Sie angeben, um wie viele Bits geschoben werden soll. Die Stellen, die durch die Schiebeoperation frei werden, werden entweder mit Nullen oder mit dem Signalzustand des Vorzeichenbits aufgefüllt ("0" steht für positiv, "1" steht für negativ). Das zuletzt geschobene Bit wird in das Bit A1 des Statusworts geladen. Die Bits A0 und OV werden auf "0" zurückgesetzt. Mit den Sprungoperationen können Sie das Bit A1 im Statuswort auswerten.

Folgende Schiebeoperationen stehen Ihnen zur Verfügung:

- SHR_I 16-Bit-Ganzzahl rechts schieben
- SHR_DI 32-Bit-Ganzzahl rechts schieben
- SHL_W 16 Bit links schieben
- SHR_W 16 Bit rechts schieben
- SHL_DW 32 Bit links schieben
- SHR_DW 32 Bit rechts schieben

11.1.2 SHR_I 16-Bit-Ganzzahl rechts schieben

Symbol

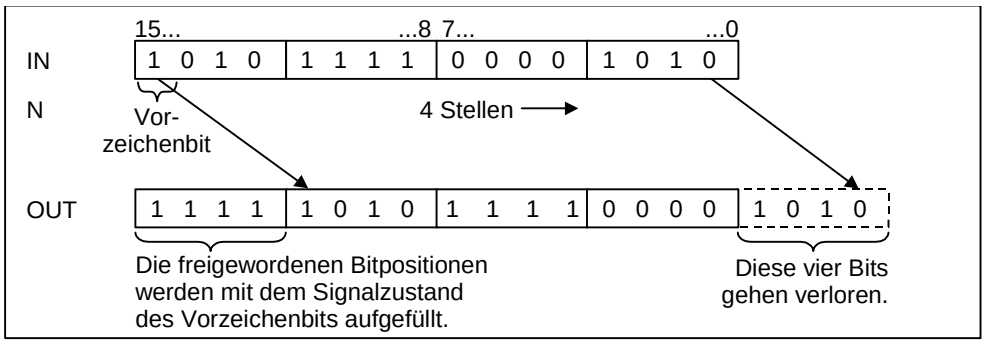


Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	INT	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die geschoben werden soll
OUT	INT	E, A, M, L, D	Ergebnis der Schiebeoperation

Beschreibung

SHR_I (16-Bit-Ganzzahl rechts schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHR_I schieben Sie die Bits 0 bis 15 von Eingang IN bitweise nach rechts. Die Bits 16 bis 31 werden nicht beeinflußt. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 16, arbeitet der Befehl so, als ob N = 16 wäre. Die Bitpositionen, die von links nachgezogen werden, um die freien Stellen zu besetzen, erhalten den Signalzustand von Bit 15 (Vorzeichenbit der Ganzzahl). Das bedeutet, daß diese Bitpositionen mit dem Wert "0" belegt werden, wenn es sich um eine positive Ganzzahl handelt, und daß sie mit dem Wert "1" belegt werden, wenn es sich um eine negative Ganzzahl handelt. Das Ergebnis der Schiebeoperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation SHR_I auf "0" gesetzt.

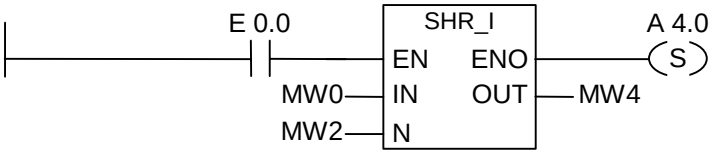
ENO hat den gleichen Signalzustand wie EN.



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

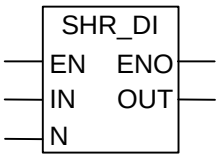
Beispiel



Die Box SHR_I wird aktiviert, wenn E 0.0 = 1 ist. MW0 wird geladen und um die Anzahl an Bits, die in MW2 angegeben ist, nach rechts verschoben. Das Ergebnis wird in MW4 geschrieben. A 4.0 wird gesetzt.

11.1.3 SHR_DI 32-Bit-Ganzzahl rechts schieben

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DINT	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die geschoben werden soll
OUT	DINT	E, A, M, L, D	Ergebnis der Schiebeoperation

Beschreibung

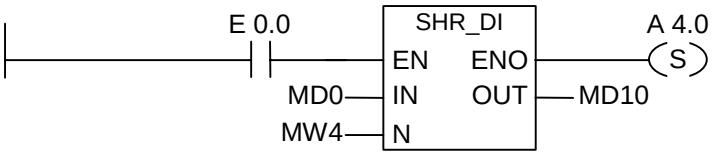
SHR_DI (32-Bit-Ganzzahl rechts schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHR_DI schieben Sie die Bits 0 bis 31 von Eingang IN bitweise nach rechts. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 32, arbeitet der Befehl so, als ob N = 32 wäre. Die Bitpositionen, die von links nachgezogen werden, um die freien Stellen zu besetzen, erhalten den Signalzustand von Bit 31 (Vorzeichenbit der Ganzzahl). Das bedeutet, daß diese Bitpositionen mit dem Wert "0" belegt werden, wenn es sich um eine positive Ganzzahl handelt, und daß sie mit dem Wert "1" belegt werden, wenn es sich um eine negative Ganzzahl handelt. Das Ergebnis der Schiebeoperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation SHR_DI auf "0" gesetzt.

ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

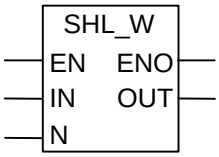
Beispiel



Die Box SHR_DI wird aktiviert, wenn E 0.0 = 1 ist. MD0 wird geladen und um die Anzahl an Bits, die in MW4 angegeben ist, nach rechts verschoben. Das Ergebnis wird in MD10 geschrieben. A 4.0 wird gesetzt.

11.1.4 SHL_W 16 Bit links schieben

Symbol

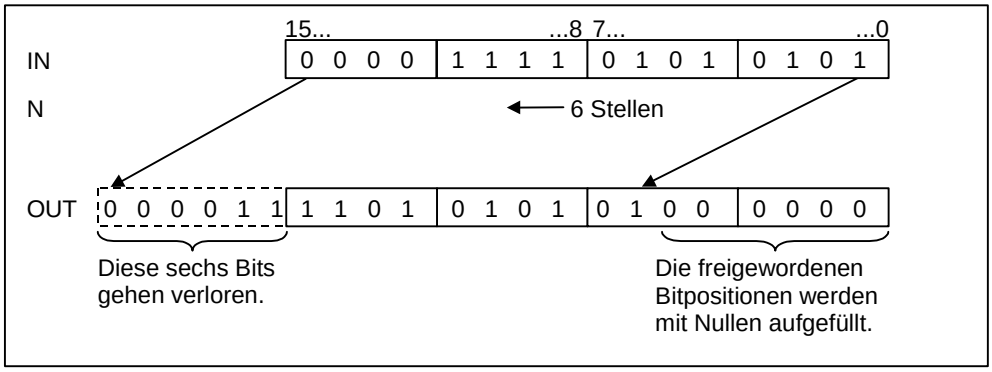


Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	WORD	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl der Bitpositionen, um die geschoben werden soll
OUT	WORD	E, A, M, L, D	Ergebnis der Schiebeoperation

Beschreibung

SHL_W (16 Bit links schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHL_W schieben Sie die Bits 0 bis 15 von Eingang IN bitweise nach links. Die Bits 16 bis 31 werden nicht beeinflußt. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 16, schreibt der Befehl in Ausgang OUT eine "0" und setzt die Bits A0 und OV des Statusworts auf "0". Von rechts wird die gleiche Anzahl (N) an Nullen geschoben, um die frei gewordenen Stellen zu belegen. Das Ergebnis der Schiebeoperation kann an Ausgang OUT abgefragt werden. Das A0-Bit und das OV-Bit werden bei N ungleich 0 von der Operation SHL_W auf "0" gesetzt.

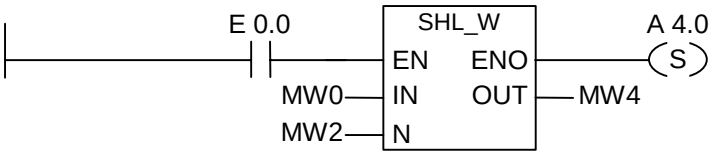
ENO hat den gleichen Signalzustand wie EN.



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

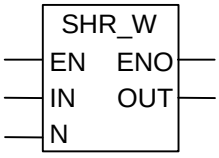
Beispiel



Die Box SHL_W wird aktiviert, wenn E 0.0 = 1 ist. MW0 wird geladen und um die Anzahl an Bits, die in MW2 angegeben ist, nach links verschoben. Das Ergebniswort wird in MW4 geschrieben. A 4.0 wird gesetzt.

11.1.5 SHR_W 16 Bit rechts schieben

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	WORD	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die geschoben werden soll
OUT	WORD	E, A, M, L, D	Ergebniswort der Schiebeoperation

Beschreibung

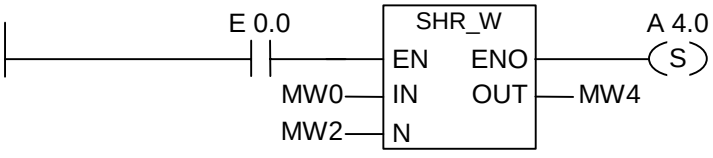
SHR_W (16 Bit rechts schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHR_W schieben Sie Bits 0 bis 15 von Eingang IN bitweise nach rechts. Die Bits 16 bis 31 werden nicht beeinflußt. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 16, schreibt der Befehl in Ausgang OUT eine "0" und setzt die Bits A0 und OV des Statusworts auf "0". Von links wird die gleiche Anzahl (N) an Nullen geschoben, um die frei gewordenen Stellen zu belegen. Das Ergebnis der Schiebeoperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation SHR_W auf "0" gesetzt.

ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

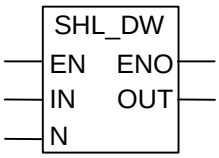
Beispiel



Die Box SHR_W wird aktiviert, wenn E 0.0 = 1 ist. MW0 wird geladen und um die Anzahl an Bits, die in MW2 angegeben ist, nach rechts verschoben. Das Ergebniswort wird in MW4 geschrieben. A 4.0 wird gesetzt.

11.1.6 SHL_DW 32 Bit links schieben

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DWORD	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die geschoben werden soll
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Schiebeoperation

Beschreibung

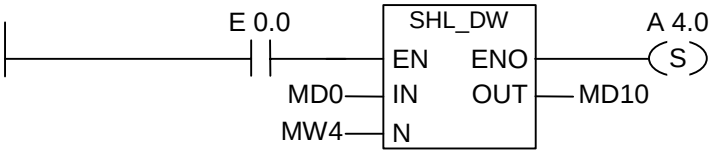
SHL_DW (32 Bit links schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHL_DW schieben Sie die Bits 0 bis 31 von Eingang IN bitweise nach links. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 32, schreibt der Befehl eine "0" in Ausgang OUT und setzt die Bits A0 und OV auf "0". Von rechts wird die gleiche Anzahl (N) an Nullen geschoben, um die frei gewordenen Stellen zu belegen. Das Ergebnisdoppelwort der Schiebeoperation kann an Ausgang OUT abgefragt werden. Das A0- und OV-Bit werden bei N ungleich 0 von der Operation SHL_DW auf "0" gesetzt.

ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

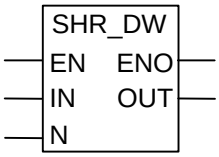
Beispiel



Die Box SHL_DW wird aktiviert, wenn E 0.0 den Signalzustand "1" hat. MD0 wird geladen und um die Anzahl an Bits, die in MW4 angegeben ist, nach links verschoben. Das Ergebnisdoppelwort wird in MD10 geschrieben. A 4.0 wird gesetzt.

11.1.7 SHR_DW 32 Bit rechts schieben

Symbol

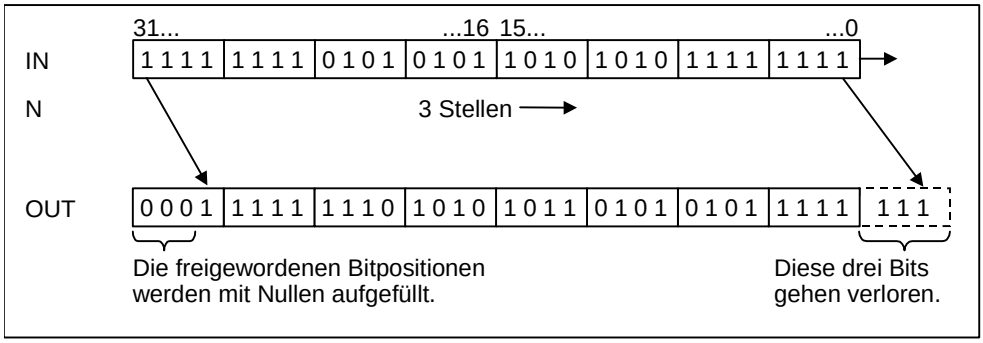


Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DWORD	E, A, M, L, D	Wert, der geschoben werden soll
N	WORD	E, A, M, L, D	Anzahl der Bitpositionen, um die geschoben werden soll
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Schiebeoperation

Beschreibung

SHR_DW (32 Bit links schieben) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation SHR_DW schieben Sie die Bits 0 bis 31 von Eingang IN bitweise nach rechts. Eingang N gibt die Anzahl der Bitpositionen an, um die geschoben werden soll. Ist N größer als 32, schreibt der Befehl eine "0" in Ausgang OUT und setzt die Bits A0 und OV auf "0". Von links wird die gleiche Anzahl (N) an Nullen geschoben, um die frei gewordenen Stellen zu belegen. Das Ergebnisdoppelwort der Schiebeoperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation SHR_DW auf "0" gesetzt.

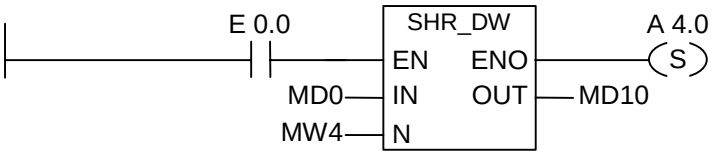
ENO hat den gleichen Signalzustand wie EN.



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

Beispiel



Die Box SHR_DW wird aktiviert, wenn E 0.0 den Signalzustand "1" hat. MD0 wird geladen und um die Anzahl an Bits, die in MW4 angegeben ist, nach rechts verschoben. Das Ergebnisdoppelwort wird in MD10 geschrieben. A 4.0 wird gesetzt.

11.2 Rotieroperationen

11.2.1 Rotieroperationen Übersicht

Beschreibung

Mit den Rotieroperationen können Sie den gesamten Inhalt von Eingang IN bitweise nach rechts oder links rotieren (siehe auch CPU-Register). Die frei gewordenen Stellen werden mit den Signalzuständen der Bits aufgefüllt, die aus dem Eingang IN geschoben wurden.

Am Eingang N können Sie angeben, um wie viele Bits rotiert werden soll.

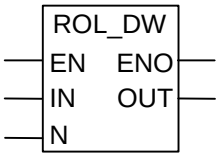
Je nach der gewählten Operation wird über das Bit A1 rotiert Das Bit A0 im Statuswort wird auf "0" zurückgesetzt.

Folgende Rotieroperationen stehen Ihnen zur Verfügung:

- ROL_DW 32 Bit links rotieren
- ROR_DW 32 Bit rechts rotieren

11.2.2 ROL_DW 32 Bit links rotieren

Symbol

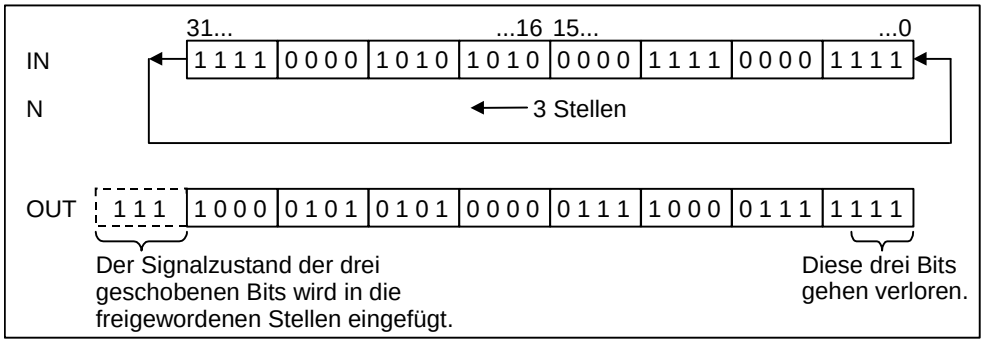


Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DWORD	E, A, M, L, D	Wert, der rotiert werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die rotiert werden soll
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Rotieroperation

Beschreibung

ROL_DW (32 Bit links rotieren) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation ROL_DW rotieren Sie den gesamten Inhalt von Eingang IN bitweise nach links. Eingang N gibt die Anzahl der Bitpositionen an, um die rotiert werden soll. Ist N größer als 32, wird das Doppelwort IN um ((N-1) modulo 32)+1 Positionen rotiert. Die Bitpositionen, die von rechts nachgezogen werden, werden mit dem Signalzustand der Bits belegt, die nach links rotiert wurden (Linksrotation). Das Ergebnisdoppelwort der Rotieroperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation ROL_DW auf "0" gesetzt.

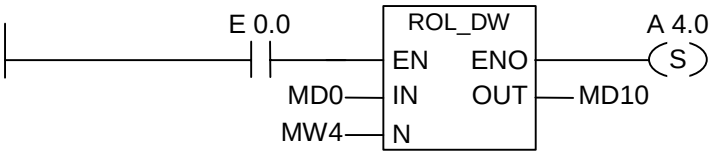
ENO hat den gleichen Signalzustand wie EN.



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

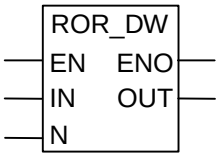
Beispiel



Die Box ROL_DW wird aktiviert, wenn E 0.0 = 1 ist. MD0 wird geladen und um die Anzahl an Bits nach links rotiert, die in MW4 angegeben ist. Das Ergebnisdoppelwort wird in MD10 geschrieben. A 4.0 wird gesetzt.

11.2.3 ROR_DW 32 Bit rechts rotieren

Symbol

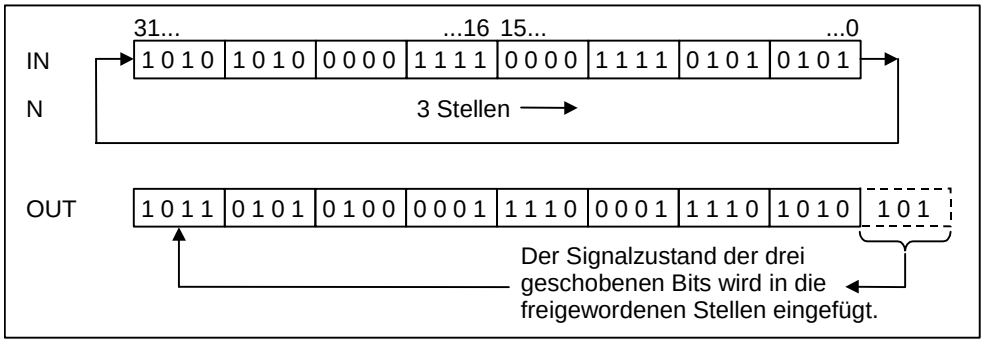


Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN	DWORD	E, A, M, L, D	Wert, der rotiert werden soll
N	WORD	E, A, M, L, D	Anzahl an Bitpositionen, um die rotiert werden soll
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Rotieroperation

Beschreibung

ROR_DW (32 Bit rechts rotieren) wird aktiviert, wenn der Freigabeeingang (EN) den Signalzustand "1" hat. Mit der Operation ROR_DW rotieren Sie den gesamten Inhalt von Eingang IN bitweise nach rechts. Eingang N gibt die Anzahl der Bitpositionen an, um die rotiert werden soll. Ist N größer als 32, wird das Doppelwort IN um ((N-1) modulo 32)+1 Positionen rotiert. Die Bitpositionen, die von links nachgezogen werden, werden mit dem Signalzustand der Bits belegt, die nach rechts rotiert wurden (Rechtsrotation). Das Ergebnisdoppelwort der Rotieroperation kann an Ausgang OUT abgefragt werden. Die A0- und OV-Bits werden bei N ungleich 0 von der Operation ROR_DW auf "0" gesetzt.

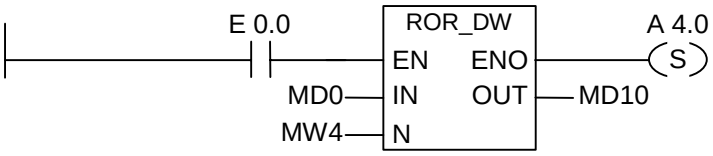
ENO hat den gleichen Signalzustand wie EN.



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	X	X	X	X	-	X	X	X	1

Beispiel



Die Box ROR_DW wird aktiviert, wenn E 0.0 = 1 ist. MD0 wird geladen und um die Anzahl an Bits nach rechts rotiert, die in MW4 angegeben ist. Das Ergebnisdoppelwort wird in MD10 geschrieben. A 4.0 wird gesetzt.

12 Statusbits

12.1 Statusbitoperationen Übersicht

Beschreibung

Statusbitoperationen sind Bitverknüpfungsoperationen, die mit den Bits des Statusworts arbeiten. Diese Operationen reagieren auf eine der folgenden Bedingungen, die von einem oder mehreren Bits angezeigt werden:

- Das Binärergebnis-Bit (BIE ---I I---) wird gesetzt (d. h. es hat einen Signalzustand von "1")
- In einer arithmetischen Operation trat ein Überlauf (UV ---I I---) oder ein gespeicherter Überlauf (OS ---I I---) auf.
- Das Ergebnis einer arithmetischen Operation ist ungültig (UO ---I I---).
- Die Beziehung des Ergebnisses einer arithmetischen Operation zu 0 ist: == 0, <> 0, > 0, < 0, >= 0, <= 0

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

Statuswort

Das Statuswort ist ein Register im Speicher Ihrer CPU. Es enthält Bits, die Sie in den Operanden von Bit- und Wortverknüpfungsoperationen ansprechen können. Aufbau des Statusworts:

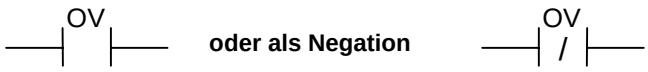
2^{15} ... ...2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER

Sie können die Bits im Statuswort auswerten

- bei Festpunkt-Funktionen,
- bei Gleitpunkt-Funktionen.

12.2 OV ---| |--- Störungsbit Überlauf

Symbol



Beschreibung

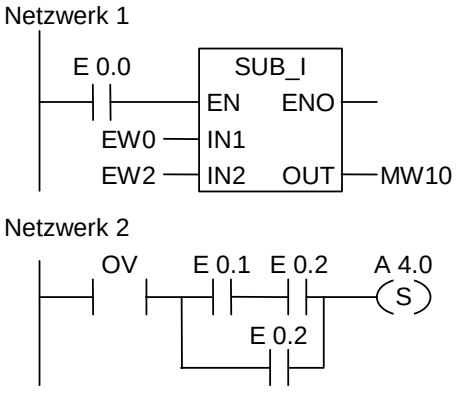
OV ---| |--- (Störungsbit Überlauf) und **OV ---| / |---** (Negiertes Störungsbit Überlauf) erkennen in der zuletzt bearbeiteten arithmetischen Operation den Überlauf. Dies bedeutet, daß sich das Ergebnis einer arithmetischen Operation außerhalb des zulässigen positiven oder negativen Bereichs befindet.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



Die Box wird von dem Zustand "1" an E 0.0 aktiviert. Wenn das Ergebnis der arithmetischen Operation EW0 - EW2 außerhalb des zulässigen Bereichs für eine Ganzzahl liegt, wird das OV-Bit gesetzt.

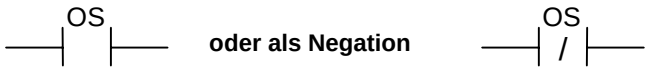
Die Signalzustandsabfrage an OV ergibt "1". Ausgang A 4.0 wird gesetzt, wenn die Abfrage bei OV "1" beträgt und das VKE von Netzwerk 2 "1" ist.

Hinweis

Die Abfrage nach Überlauf ist nur erforderlich, weil es zwei getrennte Netzwerke gibt. Andernfalls, wenn das Ergebnis außerhalb des zulässigen Bereichs liegt, ist es möglich, den Ausgang ENO der arithmetischen Operation, der den Zustand "0" hat, einzusetzen.

12.3 OS ---| |--- Störungsbit Überlauf gespeichert

Symbol



Beschreibung

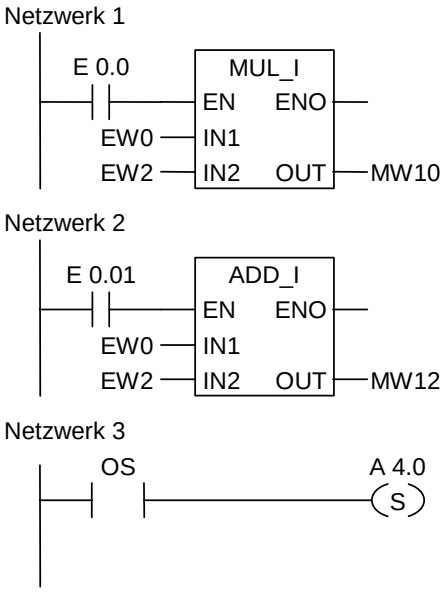
OS ---| |--- (Störungsbit Überlauf gespeichert) und **OS ---| / |---** (Negiertes Störungsbit Überlauf gespeichert) erkennen in einer arithmetischen Operation einen Überlauf und speichern diesen. Wenn sich das Ergebnis der Operation außerhalb des zulässigen positiven oder negativen Bereichs befindet, wird das OS-Bit im Statuswort gesetzt. Im Gegensatz zum OV-Bit speichert das OS-Bit einen aufgetretenen Überlauf. Das OS-Bit bleibt bis zum Verlassen des Bausteins gesetzt.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



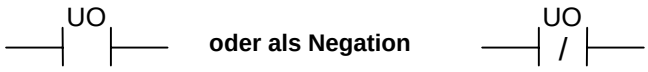
Die Box MUL_I wird von dem Zustand "1" an E 0.0 aktiviert. Die Box ADD_I wird von dem Zustand "1" an E 0.1 aktiviert. Wenn eines der beiden Ergebnisse außerhalb des zulässigen Bereichs für eine Ganzzahl liegt, wird das OS-Bit im Statuswort auf "1" gesetzt. A 4.0 wird gesetzt, wenn die Abfrage nach speicherndem Überlauf "1" ist.

Hinweis

Die Abfrage nach speicherndem Überlauf ist nur erforderlich, weil es verschiedene Netzwerke gibt. Andernfalls ist es möglich, den Ausgang ENO der ersten arithmetischen Operation an den Eingang EN der zweiten arithmetischen Operation anzuschließen (Kaskadenschaltung).

12.4 UO ---| |--- Störungsbit Ungültige Operation

Symbol



Beschreibung

UO ---| |--- (Störungsbit Ungültige Operation) und **UO ---| / |---** (Negiertes Störungsbit Ungültige Operation) erkennen, ob das Ergebnis einer arithmetischen Operation mit Gleitpunktzahlen ungültig ist (d.h. ob einer der Werte in der arithmetischen Operation keine gültige Gleitpunktzahl ist).

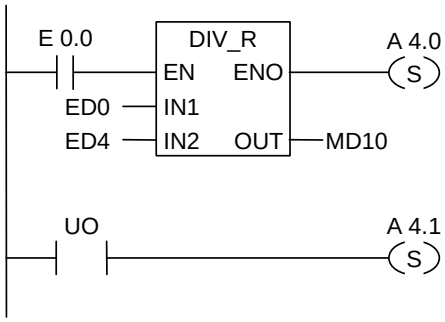
Ist das Ergebnis einer arithmetischen Operation mit Gleitpunktzahlen (UO) ungültig, so ergibt die Signalzustandabfrage 1. Gibt die Verknüpfung in A1 und A0 ”nicht ungültig” an, dann ist das Ergebnis der Signalzustandsabfrage "0".

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel

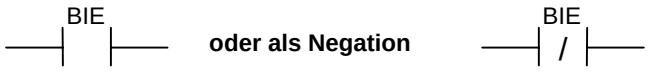


Die Box wird von dem Zustand "1" an E 0.0 aktiviert. Ist der Wert von ED0 oder ED4 keine gültige Gleitpunktzahl, so ist die arithmetische Operation ungültig. Ist der Signalzustand von EN = 1 (aktiviert) und tritt während der Bearbeitung der Funktion DIV_R ein Fehler auf, dann ist der Signalzustand von ENO = 0.

Ausgang A 4.1 wird gesetzt, wenn die Operation DIV_R ausgeführt wird, jedoch einer der Werte keine gültige **Gleitpunktzahl** ist.

12.5 BIE ---| |--- Störungsbit BIE-Register

Symbol



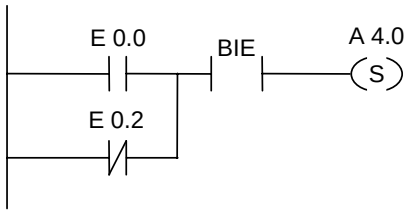
Beschreibung

BIE ---| |--- (Störungsbit BIE-Register) und **BIE ---| / |---** (Negiertes Störungsbit BIE-Register) prüfen den Zustand des BIE-Bits im Statuswort. Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft. Das BIE-Bit wird beim Übergang von Wort- zu Bitverarbeitung eingesetzt.

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	X	X	X	1

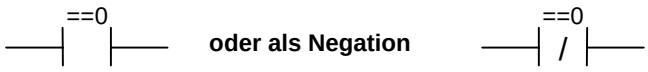
Beispiel



A 4.0 wird gesetzt, wenn E 0.0 = 1 ist ODER E 0.2 = 0 ist UND zusätzlich zu diesem VKE das BIE-Bit = 1 ist.

12.6 ==0 ---| |--- Ergebnisbit bei gleich 0

Symbol



Beschreibung

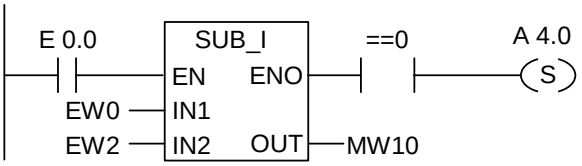
==0 ---| |--- (Ergebnisbit bei gleich 0) und **==0 ---| / |---** (Negiertes Ergebnisbit bei gleich 0) erkennen, ob das Ergebnis einer arithmetischen Operation gleich "0" ist. Die Operationen fragen die Anzeigenbits A1 und A0 im Statuswort ab, um die Beziehung des Ergebnisses zu "0" zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

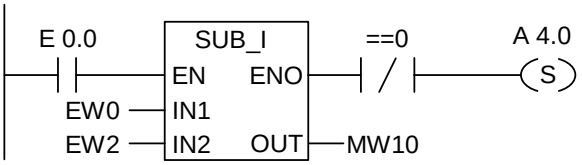
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiele



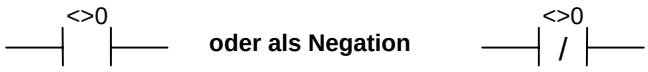
Die Operation wird von dem Zustand "1" an E 0.0 aktiviert. Ist der Wert von EW0 gleich dem Wert von EW2, dann ist das Ergebnis der arithmetischen Operation EW0 - EW2 gleich "0". A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis gleich "0" ist.



A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis nicht gleich "0" ist.

12.7 <>0 ---| |--- Ergebnisbit bei ungleich 0

Symbol



Beschreibung

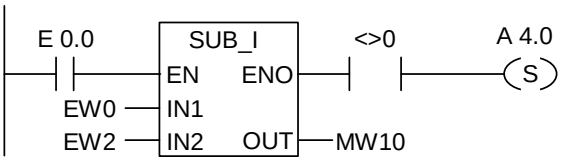
<>0 ---| |--- (Ergebnisbit bei ungleich 0) und **<>0 ---| / |---** (Negiertes Ergebnisbit bei ungleich 0) erkennen, ob das Ergebnis einer arithmetischen Operation ungleich "0" ist. Die Operationen fragen die Anzeigebits A1 und A0 im Statuswort ab, um die Beziehung des Ergebnisses zu "0" zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

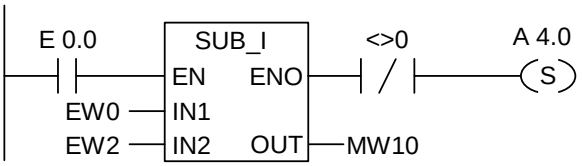
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiele



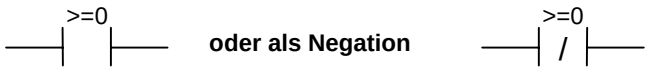
Die Operation wird von dem Zustand "1" an E 0.0 aktiviert. Ist der Wert von EW0 ungleich dem Wert von EW2,dann ist das Ergebnis der arithmetischen Operation EW0 - EW2 ungleich "0". A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis ungleich "0" ist.



A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis gleich "0" ist.

12.8 >=0 ---| |--- Ergebnisbit bei größer gleich 0

Symbol



Beschreibung

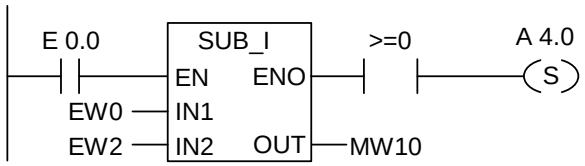
>=0 ---| |--- (Ergebnisbit bei größer gleich 0) und **>=0 ---| / |---** (Negiertes Ergebnisbit bei größer gleich 0) erkennen, ob das Ergebnis einer arithmetischen Operation größer als oder gleich "0" ist. Die Operationen fragen die Anzeigebits A1 und A0 im Statuswort ab, um die Beziehung zu "0" zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

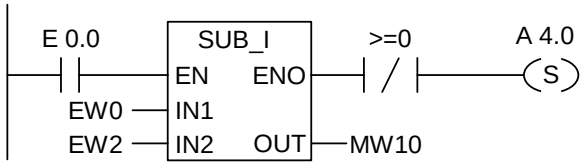
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiele



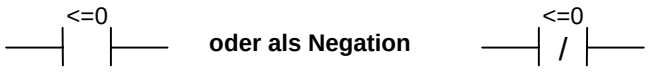
Die Operation wird von dem Zustand "1" an E 0.0 aktiviert. Ist der Wert von EW0 größer als oder gleich dem Wert von EW2, dann ist das Ergebnis der arithmetischen Operation EW0 - EW2 größer als oder gleich "0". A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis größer gleich "0" ist.



A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis nicht größer gleich "0" ist.

12.9 <=0 ---| |--- Ergebnisbit bei kleiner gleich 0

Symbol



Beschreibung

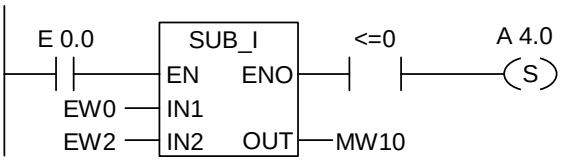
<=0 ---| |--- (Ergebnisbit bei kleiner gleich 0) und **<=0 ---| / |---** (Negiertes Ergebnisbit bei kleiner gleich 0) erkennen, ob das Ergebnis einer arithmetischen Operation kleiner als oder gleich "0" ist. Die Operationen fragen die Anzeigebits A1 und A0 im Statuswort ab, um die Beziehung des Ergebnisses zu 0 zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis durch UND mit dem VKE verknüpft, bei Parallelschaltungen ist das Ergebnis durch ODER mit dem VKE verknüpft.

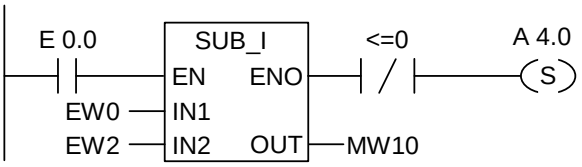
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiele



Die Operation wird von dem Zustand "1" an E 0.0 aktiviert. Ist der Wert von EW0 kleiner als oder gleich dem Wert von EW2, dann ist das Ergebnis der arithmetischen Operation EW0 - EW2 kleiner als oder gleich "0". A 4.0 wird gesetzt, wenn die Funktion fehlerfrei ausgeführt wird und das Ergebnis kleiner gleich "0" ist.



A 4.0 wird gesetzt, wenn die Funktion fehlerfrei ausgeführt wird und das Ergebnis nicht kleiner gleich "0" ist.

12.10 >0 ---| |--- Ergebnisbit bei größer als 0

Symbol

Beschreibung

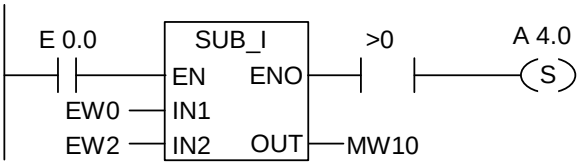
>0 ---| |--- (Ergebnisbit bei größer als 0) und **>0 ---| / |---** (Negiertes Ergebnisbit bei größer als 0) erkennen, ob das Ergebnisse einer arithmetischen Operation größer als 0 ist. Die Operationen fragen die Anzeigebits A1 und A0 ab, um die Beziehung zu "0" zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis mit dem VKE durch UND verknüpft, bei Parallelschaltungen ist das Ergebnis mit dem VKE durch ODER verknüpft.

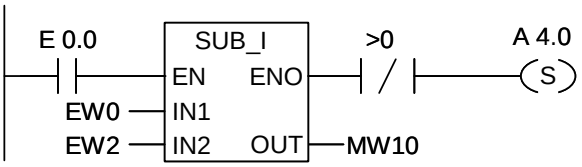
Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



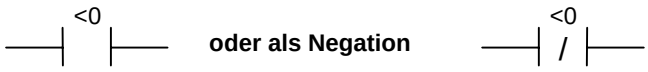
Die Box wird von dem Zustand "1" an E 0.0 aktiviert. Wenn der Wert von EW0 größer ist als der Wert von EW2, ist das Ergebnis der arithmetischen Operation EW0 - EW2 größer als "0". A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis größer als "0" ist .



A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis nicht größer als "0" ist.

12.11 <0 ---| |--- Ergebnisbit bei kleiner 0

Symbol



Beschreibung

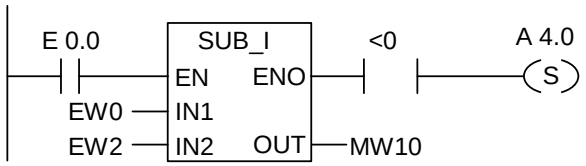
<0 ---| |--- (Ergebnisbit bei kleiner als 0) und **<0 ---| / |---** (Negiertes Ergebnisbit bei kleiner als 0) erkennen, ob das Ergebnis einer arithmetischen Operation kleiner als "0" ist. Die Operationen fragen die Anzeigebits A1 und A0 im Statuswort ab, um die Beziehung des Ergebnisses zu "0" zu bestimmen.

Bei Reihenschaltungen ist das Abfrageergebnis mit dem VKE durch UND verknüpft, bei Parallelschaltungen ist das Ergebnis mit dem VKE durch ODER verknüpft.

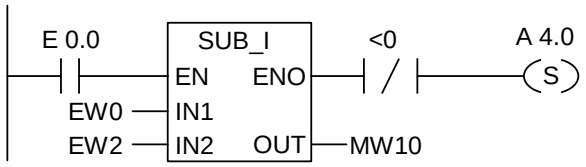
Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiele



Die Operation wird von dem Zustand "1" an E 0.0 aktiviert. Wenn der Wert von EW0 kleiner ist als der Wert von EW2, ist das Ergebnis der arithmetischen Operation EW0 - EW2 kleiner als "0". A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis kleiner als "0" ist.



A 4.0 wird gesetzt, wenn die Operation fehlerfrei ausgeführt wird und das Ergebnis nicht kleiner als "0" ist.

13 Zeiten

13.1 Zeitoperationen Übersicht

Beschreibung

Unter "Speicherbereiche und Komponenten einer Zeit" finden Sie Informationen zum Einstellen und zur Auswahl der richtigen Zeit.

Folgende Zeitoperationen stehen Ihnen zur Verfügung:

- S_IMPULS Zeit als Impuls parametrieren und starten
- S_VIMP Zeit als verlängerten Impuls parametrieren und starten
- S_EVERZ Zeit als Einschaltverzögerung parametrieren und starten
- S_SEVERZ Zeit als speichernde Einschaltverzögerung parametrieren und starten
- S_AVERZ Zeit als Ausschaltverzögerung parametrieren und starten

- ---(SI) Zeit als Impuls starten
- ---(SV) Zeit als verlängerten Impuls starten
- ---(SE) Zeit als Einschaltverzögerung starten
- ---(SS) Zeit als speichernde Einschaltverzögerung starten
- ---(SA) Zeit als Ausschaltverzögerung starten

13.2 Speicherbereiche und Komponenten einer Zeit

Speicherbereich

Zeiten haben einen eigenen reservierten Speicherbereich in Ihrer CPU. Dieser Speicherbereich reserviert ein 16-Bit-Wort für jeden Zeitoperanden. Das Programmieren mit KOP unterstützt 256 Zeiten. Wie viele Zeitworte in Ihrer CPU zur Verfügung stehen, entnehmen Sie bitte deren technischen Daten.

Folgende Funktionen greifen auf den Speicherbereich der Zeiten zu:

- Zeitoperationen
- Aktualisieren der Timerwörter über Zeitimpulsgeber. Diese Funktion Ihrer CPU im RUN-Zustand vermindert einen bestimmten Wert um jeweils eine Einheit in einem Intervall, das von der Zeitbasis festgelegt wurde, bis der Zeitwert gleich "0" ist.

Zeitwert

Die Bits 0 bis 9 des Timerworts enthalten den Zeitwert binär-codiert. Der Zeitwert gibt eine Anzahl von Einheiten an. Das Aktualisieren der Zeit vermindert den Zeitwert um jeweils eine Einheit in einem Intervall, der von der Zeitbasis festgelegt wurde. Der Zeitwert wird solange vermindert, bis er gleich "0" ist. Sie können einen Zeitwert im dualen, hexadezimalen oder binär-codierten Dezimalformat (BCD) laden.

Mit der folgenden Syntax können Sie einen vordefinierten Zeitwert laden:

- w#16#wxyz
  - w = Zeitbasis (d. h. Zeitintervall oder Auflösung)
  - xyz = Zeitwert im BCD-Format
- S5T#a**H**_b**M**_c**S**_d**MS**
  - H (Stunden), M (Minuten), S (Sekunden), MS (Millisekunden); a, b, c, d werden vom Anwender definiert
  - Die Zeitbasis wird automatisch gewählt und der Wert zur nächstniederen Zahl mit dieser Zeitbasis gerundet

Sie können einen Zeitwert von max. 9 990 Sekunden bzw. 2H_46M_30S eingeben. Beispiele:

S5TIME#4S = 4 Sekunden
s5t#2h_15m = 2 Stunden und 15 Minuten
S5T#1H_12M_18S = 1 Stunde, 12 Minuten und 18 Sekunden

Zeitbasis

Die Bits 12 und 13 des Timerworts enthalten die Zeitbasis binär-codiert. Die Zeitbasis definiert das Intervall, in dem der Zeitwert um eine Einheit vermindert wird. Die kleinste Zeitbasis beträgt 10 ms, die größte 10 s.

Zeitbasis	Binärcode für Zeitbasis
10 ms	00
100 ms	01
1 s	10
10 s	11

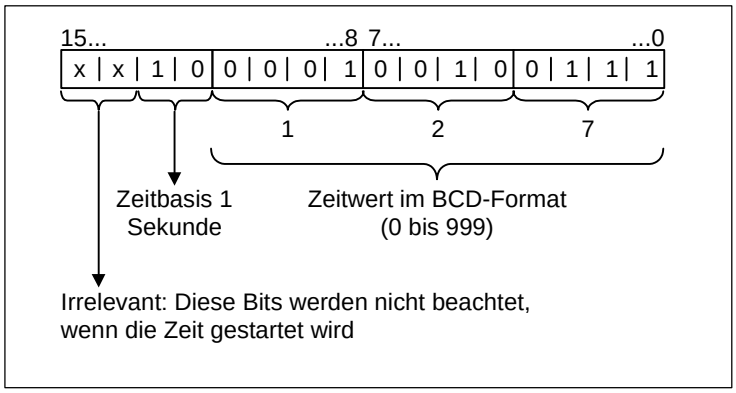
Die Werte dürfen 2H_46M_30S nicht überschreiten. Werte, die für einen Bereich oder für eine Auflösung zu groß sind, werden gerundet. Das allgemeine Format für den Datentyp S5TIME hat folgende Grenzwerte:

Auflösung	Bereich
0,01 Sekunde	10MS bis 9S_990MS
0,1 Sekunde	100MS bis 1M_39S_900MS
1 Sekunde	1S bis 16M_39S
10 Sekunden	10S bis 2H_46M_30S

Bit-Konfiguration in der Zeitzelle

Wird eine Zeit gestartet, so wird der Inhalt der Zeitzelle als Zeitwert verwendet. Die Bits 0 bis 11 der Zeitzelle enthalten den Zeitwert im binär-codierten Dezimalformat (BCD-Format: jede Gruppe von vier Bits enthält den Binärcode für einen Dezimalwert). Die Bits 12 und 13 enthalten die Zeitbasis im Binärcode.

Folgendes Bild zeigt den Inhalt der Zeitzelle, nachdem Sie den Zeitwert 127 mit der Zeitbasis 1 Sekunde geladen haben:

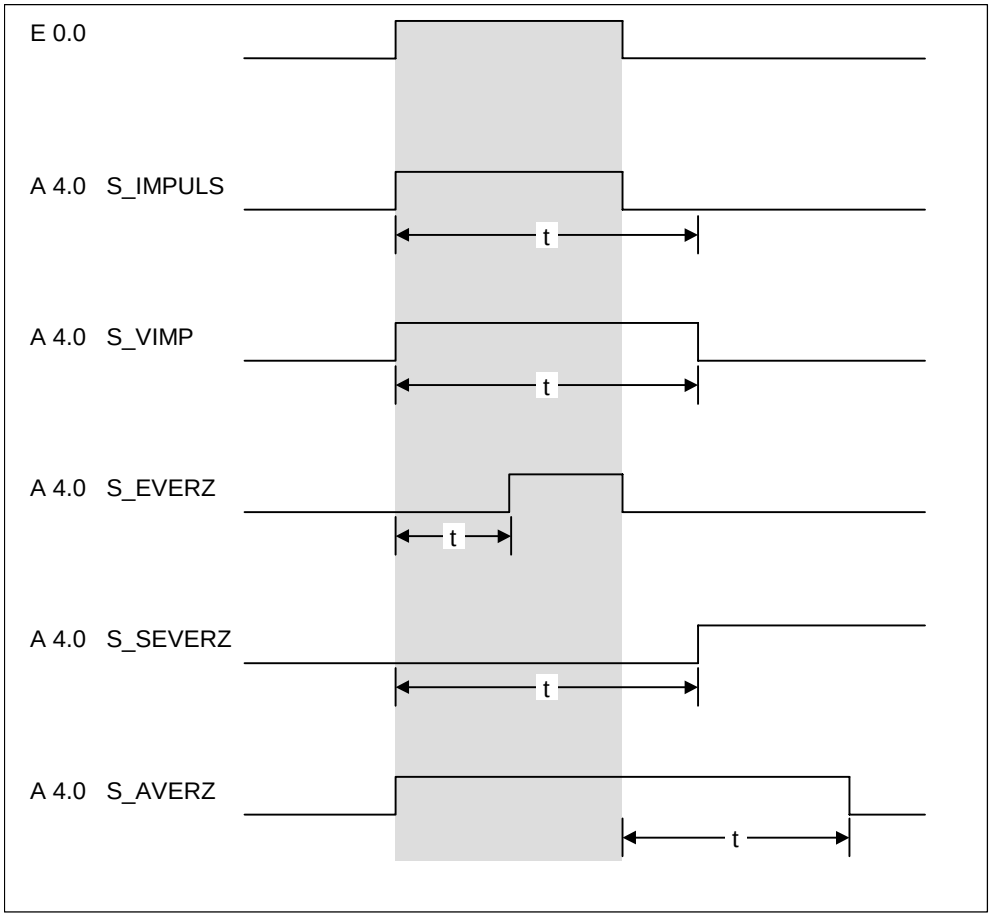


Lesen der Zeit und der Zeitbasis

Jede Timerbox liefert zwei Ausgänge, DUAL und DEZ, für die Sie eine Wortadresse angeben können. Am Ausgang DUAL ist der Zeitwert binär-codiert, die Zeitbasis wird nicht angezeigt. Am Ausgang DEZ sind Zeitbasis und Zeitwort BCD-codiert.

Auswahl der richtigen Zeit

Die Übersicht über die 5 verschiedenen Zeiten soll Ihnen helfen, die für Ihre Zwecke adäquate Zeit auszuwählen.

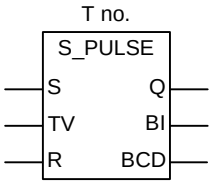


Zeiten	Erklärung
S_IMPULS Zeit als Impuls	Die maximale Zeit, in der das Ausgangssignal auf "1" bleibt, ist gleich dem programmierten Zeitwert t. Das Ausgangssignal bleibt für eine kürzere Zeit auf "1", wenn das Eingangssignal auf "0" wechselt.
S_VIMP Zeit als verlängerter Impuls	Das Ausgangssignal bleibt für die programmierte Zeit auf "1", unabhängig davon, wie lange das Eingangssignal auf "1" bleibt.
S_EVERZ Zeit als Einschaltverzögerung	Das Ausgangssignal ist nur "1", wenn die programmierte Zeit abgelaufen ist und das Eingangssignal noch immer "1" beträgt.
S_SEVERZ Zeit als speichernde Einschaltverzögerung	Das Ausgangssignal wechselt nur von "0" auf "1", wenn die programmierte Zeit abgelaufen ist, unabhängig davon, wie lange das Eingangssignal auf "1" bleibt.
S_AVERZ Zeit als Ausschaltverzögerung	Das Ausgangssignal ist "1", wenn das Eingangssignal "1" ist oder die Zeit läuft. Die Zeit wird gestartet wenn das Eingangssignal von "1" auf "0" wechselt.

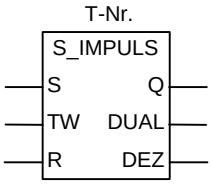
13.3 S_IMPULS Zeit als Impuls parametrieren und starten

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
T no.	T-Nr.	TIMER	T	Nummer der Zeit, Anzahl der Zeiten ist von der CPU abhängig
S	S	BOOL	E, A, M, L, D	Starteingang
TV	TW	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
BI	DUAL	WORD	E, A, M, L, D	Aktueller Zeitwert, binär-codiert
BCD	DEZ	WORD	E, A, M, L, D	Aktueller Zeitwert, BCD-Format
Q	Q	BOOL	E, A, M, L, D	Status der Zeit

Beschreibung

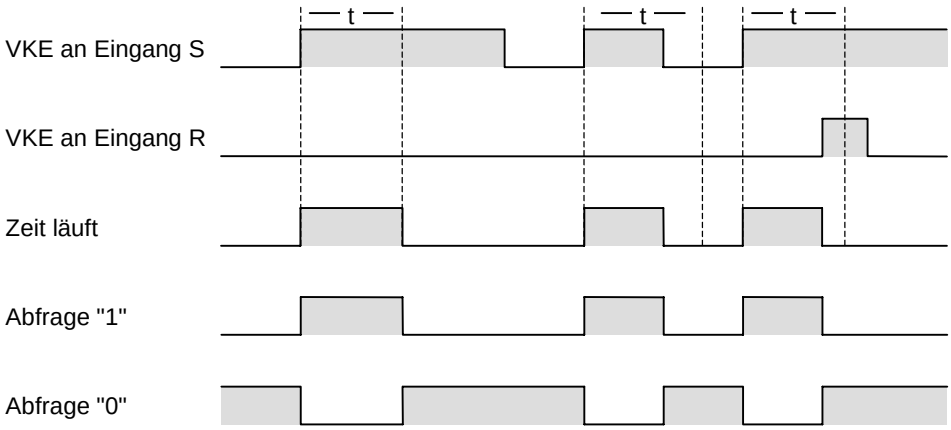
S_IMPULS (Zeit als Impuls parametrieren und starten) startet die angegebene Zeit bei einer steigenden Flanke am Starteingang S. Es ist immer ein Signalwechsel erforderlich, um eine Zeit zu starten. Die Zeit läuft, solange der Signalzustand am Eingang S "1" ist, längstens jedoch für die Dauer des an Eingang TW angegebenen Zeitwerts. Der Signalzustand am Ausgang Q ist "1", solange die Zeit läuft. Wechselt der Signalzustand an Eingang S von "1" auf "0", bevor das Zeitintervall abgelaufen ist, wird die Zeit angehalten. In diesem Fall ist der Signalzustand am Ausgang Q "0".

Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang R der Zeit von "0" auf "1" geht, während die Zeit läuft. Der aktuelle Zeitwert und die Zeitbasis werden auch auf Null gesetzt. Eine "1" am Eingang R der Zeit hat keine Auswirkungen, wenn die Zeit nicht läuft.

Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert am Ausgang DUAL ist binär-codiert, der Wert am Ausgang DEZ ist BCD-codiert. Der aktuelle Zeitwert entspricht dem Anfangswert von TW, von dem der Zeitwert abgezogen wird, der seit dem Start der Zeit vergangen ist.

Siehe auch "Speicherbereiche und Komponenten einer Zeit".

Impulsdiagramm

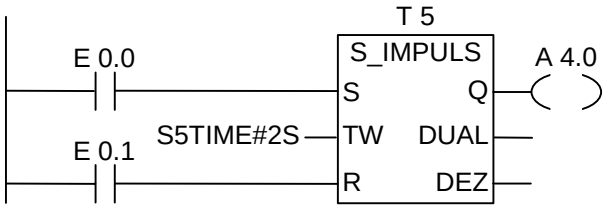


t = programmierte Zeit

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



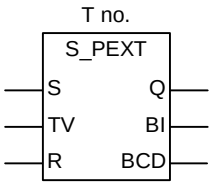
Wechselt der Signalzustand des Eingangs E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Die Zeit läuft mit dem angegebenen Zeitwert von zwei Sekunden (2 s) solange Eingang E 0.0 = 1 ist. Wechselt der Signalzustand von E 0.0 von "1" auf "0", bevor die Zeit abgelaufen ist, wird die Zeit gestoppt. Wechselt der Signalzustand des Eingangs E 0.1 von "0" auf "1" während die Zeit läuft, wird die zeit zurückgesetzt.

Ausgang A 4.0 ist "1", solange die Zeit läuft und "0", wenn die Zeit abgelaufen ist oder zurückgesetzt wurde.

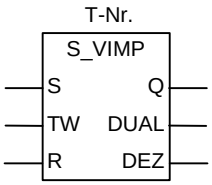
13.4 S_VIMP Zeit als verlängerten Impuls parametrieren und starten

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
T no.	T-Nr.	TIMER	T	Nummer der Zeit, Anzahl der Zeiten ist von der CPU abhängig
S	S	BOOL	E, A, M, L, D	Starteingang
TV	TW	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
BI	DUAL	WORD	E, A, M, L, D	Aktueller Zeitwert, binär-codiert
BCD	DEZ	WORD	E, A, M, L, D	Aktueller Zeitwert, BCD-Format
Q	Q	BOOL	E, A, M, L, D	Status der Zeit

Beschreibung

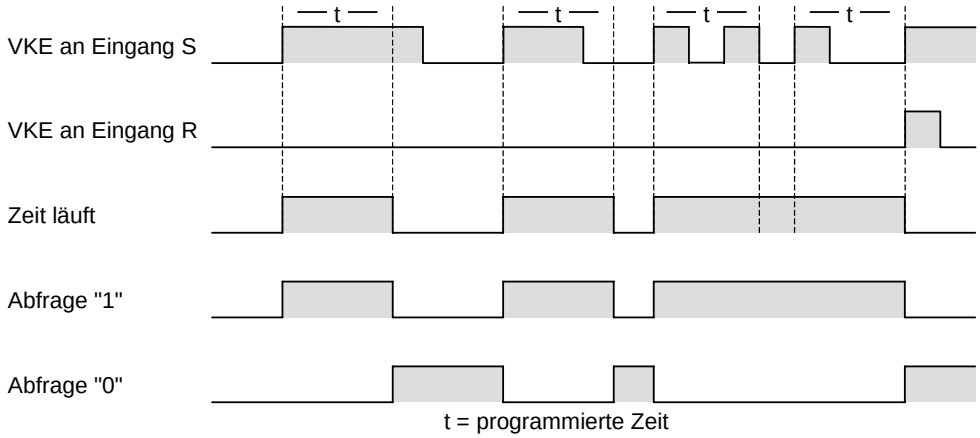
S_VIMP (Zeit als verlängerten Impuls parametrieren und starten) startet die angegebene Zeit bei einer steigenden Flanke am Starteingang S. Es ist immer ein Signalwechsel erforderlich, um eine Zeit zu starten. Die Zeit läuft für die Dauer der an Eingang TW angegebenen, voreingestellten Zeit, auch wenn der Signalzustand an Eingang S auf "0" geht, bevor das Zeitintervall abgelaufen ist. Der Signalzustand am Ausgang Q ist "1", solange die Zeit läuft. Die Zeit wird mit dem voreingestellten Zeitwert neu gestartet, wenn der Signalzustand am Eingang S von "0" auf "1" wechselt, während die Zeit läuft.

Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang R der Zeit von "0" auf "1" geht, während die Zeit läuft. Der aktuelle Zeitwert und die Zeitbasis werden auf Null gesetzt.

Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert am Ausgang DUAL ist binär-codiert, der Wert am Ausgang DEZ ist BCD-codiert. Der aktuelle Zeitwert entspricht dem Anfangswert von TW, von dem der Zeitwert abgezogen wird, der seit dem Start der Zeit vergangen ist.

Siehe auch "Speicherbereiche und Komponenten einer Zeit".

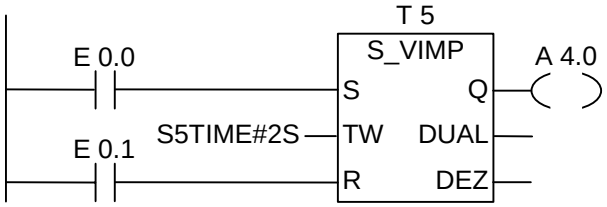
Impulsdiagramm



Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel

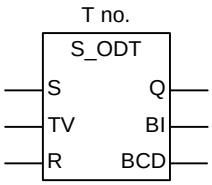


Wechselt der Signalzustand an E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Die Zeit läuft mit dem angegebenen Zeitwert von zwei Sekunden weiter, ohne von einer fallenden Flanke am Eingang S beeinträchtigt zu werden. Wechselt der Signalzustand von E 0.0 von "0" auf "1", bevor die Zeit abgelaufen ist, wird die Zeit neu gestartet. Wechselt der Signalzustand des Eingangs E 0.1 von "0" auf "1" während die Zeit läuft, wird die Zeit zurückgesetzt. Ausgang A 4.0 ist "1", solange die Zeit läuft.

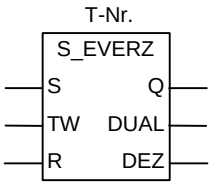
13.5 S_EVERZ Zeit als Einschaltverzögerung parametrieren und starten

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
T no.	T-Nr.	TIMER	T	Nummer der Zeit, Anzahl der Zeiten ist von der CPU abhängig
S	S	BOOL	E, A, M, L, D	Starteingang
TV	TW	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
BI	DUAL	WORD	E, A, M, L, D	Aktueller Zeitwert, binär-codiert
BCD	DEZ	WORD	E, A, M, L, D	Aktueller Zeitwert, BCD-Format
Q	Q	BOOL	E, A, M, L, D	Status der Zeit

Beschreibung

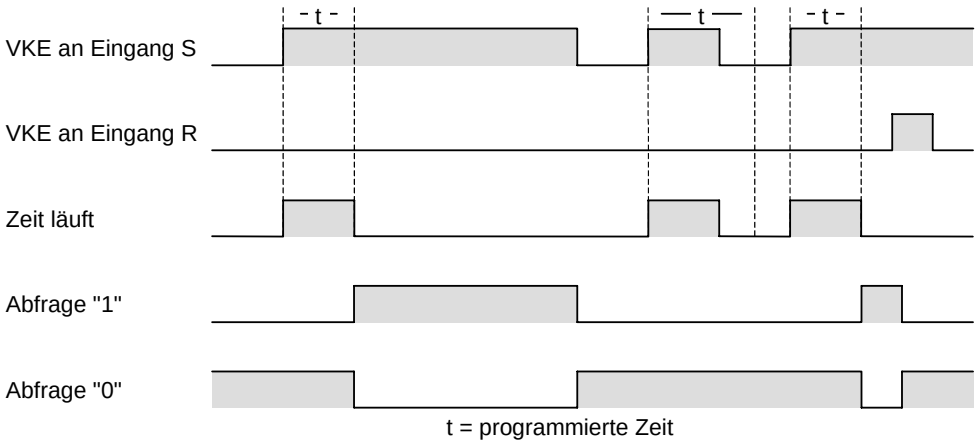
S_EVERZ (Zeit als Einschaltverzögerung parametrieren und starten) startet die angegebene Zeit bei einer steigenden Flanke am Starteingang S. Ein Signalwechsel ist immer erforderlich, um eine Zeit zu starten. Die Zeit läuft mit dem an Eingang TW angegebenen Zeitwert solange weiter, wie der Signalzustand an Eingang S positiv ist. Der Signalzustand am Ausgang Q ist "1", wenn die Zeit fehlerfrei abgelaufen ist und der Signalzustand des Eingangs S "1" ist. Wechselt der Signalzustand an Eingang S von "1" auf "0", während die Zeit läuft, wird die Zeit angehalten. In diesem Fall ist der Signalzustand am Ausgang Q "0".

Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang R der Zeit von "0" auf "1" geht, während die Zeit läuft. Der Zeitwert und die Zeitbasis werden ebenfalls auf Null gesetzt. Der Signalzustand am Ausgang Q ist dann "0". Die Zeit wird auch zurückgesetzt, wenn am Rücksetzeingang R der Wert "1" anliegt, während die Zeit nicht läuft und das VKE am Eingang S "1" ist.

Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert am Ausgang DUAL ist binärcodiert, der Wert am Ausgang DEZ ist BCD-codiert. Der aktuelle Zeitwert entspricht dem Anfangswert von TW, von dem der Zeitwert abgezogen wird, der seit dem Start der Zeit abgelaufen ist.

Siehe auch "Speicherbereiche und Komponenten einer Zeit".

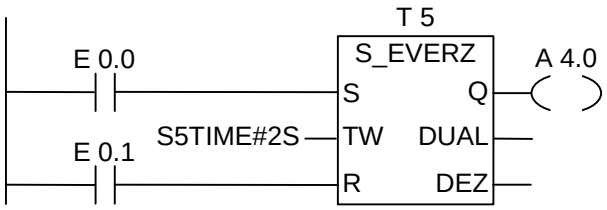
Impulsdiagramm



Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel

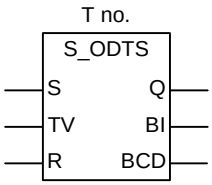


Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Läuft die Zeit von zwei Sekunden ab und der Signalzustand an Eingang E 0.0 ist noch immer "1", dann ist Ausgang A 4.0 "1". Wechselt der Signalzustand von E 0.0 von "1" auf "0", dann wird die Zeit angehalten, und Ausgang A 4.0 ist "0" (Wechselt der Signalzustand des Eingangs E 0.1 von "0" auf "1", wird die Zeit zurückgesetzt, unabhängig davon, ob die Zeit läuft oder nicht).

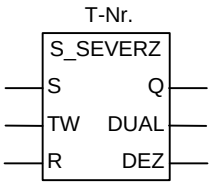
13.6 S_SEVERZ Zeit als speichernde Einschaltverzögerung parametrieren und starten

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
T no.	T-Nr.	TIMER	T	Nummer der Zeit, Anzahl der Zeiten ist von der CPU abhängig
S	S	BOOL	E, A, M, L, D	Starteingang
TV	TW	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
BI	DUAL	WORD	E, A, M, L, D	Aktueller Zeitwert, binär-codiert
BCD	DEZ	WORD	E, A, M, L, D	Aktueller Zeitwert, BCD-Format
Q	Q	BOOL	E, A, M, L, D	Status der Zeit

Beschreibung

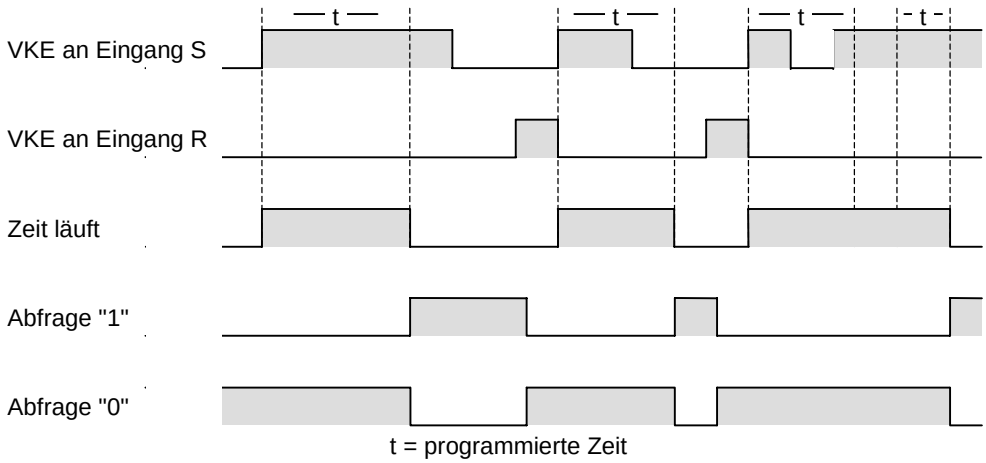
S_SEVERZ (Zeit als speichernde Einschaltverzögerung parametrieren und starten) startet die angegebene Zeit bei einer steigenden Flanke am Starteingang S. Ein Signalwechsel ist immer erforderlich, um eine Zeit zu starten. Die Zeit läuft mit dem an Eingang TW angegebenen Zeitwert auch dann weiter, wenn der Signalzustand an Eingang S auf "0" geht, bevor die Zeit abgelaufen ist. Der Signalzustand am Ausgang Q ist "1", wenn die Zeit abgelaufen ist, unabhängig vom Signalzustand am Eingang S. Die Zeit wird mit dem angegebenen Zeitwert neu gestartet, wenn der Signalzustand an Eingang S von "0" auf "1" wechselt, während die Zeit läuft.

Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang R der Zeit von "0" auf "1" geht, unabhängig vom VKE am Eingang S. Der Signalzustand am Ausgang Q ist dann "0".

Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert am Ausgang DUAL ist binär-codiert, der Wert am Ausgang DEZ ist BCD-codiert. Der aktuelle Zeitwert entspricht dem Anfangswert von TW, von dem der Zeitwert abgezogen wird, der seit dem Start der Zeit vergangen ist.

Siehe auch "Speicherbereiche und Komponenten einer Zeit".

Impulsdiagramm

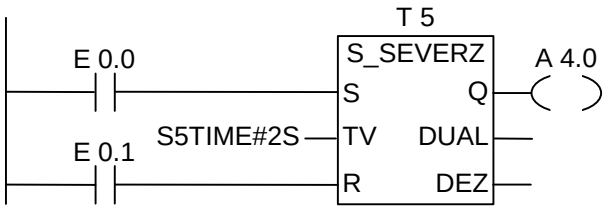


t = programmierte Zeit

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel

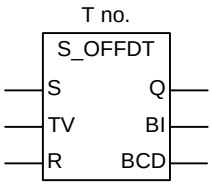


Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Die Zeit läuft, ohne von einem Signalwechsel von "1" auf "0" an Eingang E 0.0 beeinflußt zu werden. Wechselt der Signalzustand von E 0.0 von "0" auf "1", bevor die Zeit abgelaufen ist, dann wird die Zeit neu gestartet. Ausgang A 4.0 ist "1", wenn die Zeit abgelaufen ist (Wechselt der Signalzustand des Eingangs E 0.1 von "0" auf "1", wird die Zeit zurückgesetzt, unabhängig von VKE an S).

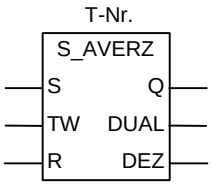
13.7 S_AVERZ Zeit als Ausschaltverzögerung parametrieren und starten

Symbol

Englisch



Deutsch



Parameter Englisch	Parameter Deutsch	Datentyp	Speicher-bereich	Beschreibung
T no.	T-Nr.	TIMER	T	Nummer der Zeit, Anzahl der Zeiten ist von der CPU abhängig
S	S	BOOL	E, A, M, L, D	Starteingang
TV	TW	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert
R	R	BOOL	E, A, M, L, D	Rücksetzeingang
BI	DUAL	WORD	E, A, M, L, D	Aktueller Zeitwert, binär-codiert
BCD	DEZ	WORD	E, A, M, L, D	Aktueller Zeitwert, BCD-Format
Q	Q	BOOL	E, A, M, L, D	Status der Zeit

Beschreibung

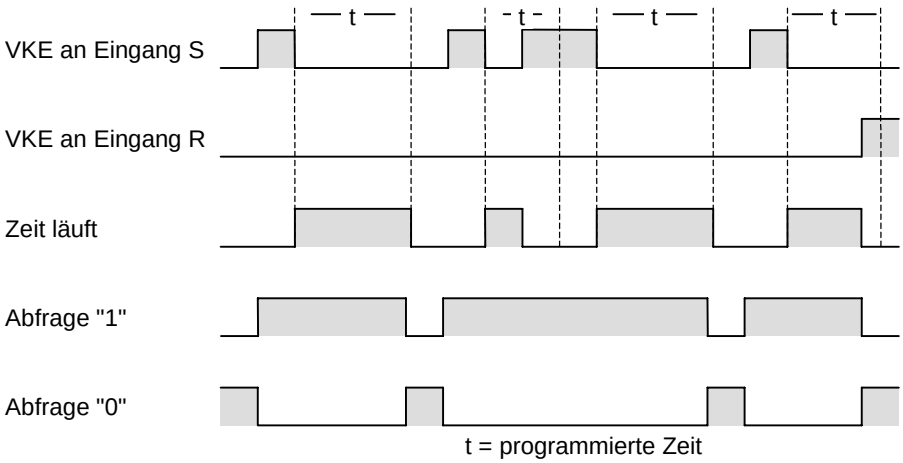
S_AVERZ (Zeit als Ausschaltverzögerung parametrieren und starten) startet die angegebene Zeit bei einer fallenden Flanke am Starteingang S. Ein Signalwechsel ist immer erforderlich, um eine Zeit zu starten. Der Signalzustand am Ausgang Q ist "1", wenn der Signalzustand am Eingang S "1" ist oder während die Zeit läuft. Die Zeit wird zurückgesetzt, wenn der Signalzustand an Eingang S von "0" auf "1" wechselt, während die Zeit läuft. Die Zeit wird erst dann neu gestartet, wenn der Signalzustand am Eingang S wieder von "1" auf "0" wechselt.

Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang R von "0" auf "1" geht, während die Zeit läuft.

Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert am Ausgang DUAL ist binär-codiert, der Wert am Ausgang DEZ ist BCD-codiert. Der aktuelle Zeitwert entspricht dem Anfangswert von TW, von der der Zeitwert abgezogen wird, der seit dem Start der Zeit vergangen ist.

Siehe auch "Speicherbereiche und Komponenten einer Zeit".

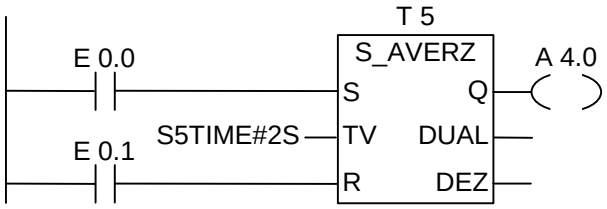
Impulsdiagramm



Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	X	X	X	1

Beispiel



Die Zeit wird gestartet, wenn der Signalzustand an Eingang E 0.0 von "1" auf "0" wechselt.

A 4.0 ist "1", wenn E 0.0 "1" ist oder die Zeit läuft (Wechselt der Signalzustand an E 0.1 von "0" auf "1", während die Zeit läuft, wird die Zeit zurückgesetzt).

13.8 ---(SI) Zeit als Impuls starten

Symbol

Englisch	Deutsch
<T-Nr.>	<T-Nr.>
---(SP)	**---(SI)**
<Zeitwert>	<Zeitwert>

Parameter	Datentyp	Speicherbereich	Beschreibung
<T-Nr.>	TIMER	T	Nummer der Zeit, Anzahl der Zeiten hängt von der CPU ab
<Zeitwert>	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert

Beschreibung

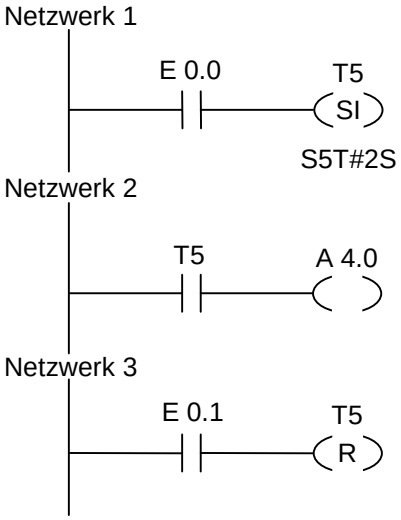
---(SI) (Zeit als Impuls starten) startet die angegebene Zeit mit dem **<Zeitwert>**, wenn im VKE eine steigende Flanke auftritt. Die Zeit läuft so lange mit dem angegebenen Zeitintervall, wie das VKE positiv ist ("1"). Eine Signalzustandsabfrage nach "1" ergibt "1", solange die Zeit läuft. Wechselt das VKE von "1" auf "0", bevor der Zeitwert abgelaufen ist, wird die Zeit gestoppt. In diesem Fall ergibt die Signalzustandsabfrage nach "1" das Ergebnis "0".

Siehe auch "Speicherbereiche und Komponenten einer Zeit" und S_IMPULS (Zeit als Impuls starten).

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Die Zeit läuft mit dem angegebenen Zeitwert von 2 s so lange weiter, wie E 0.0 "1" ist. Wechselt der Signalzustand an E 0.0 von "1" auf "0", bevor die Zeit abgelaufen ist, wird die Zeit gestoppt. Ausgang A 4.0 ist "1", solange die Zeit läuft. Wechselt der Signalzustand an Eingang E 0.1 von "0" auf "1", dann wird die Zeit T5 zurückgesetzt, d.h. sie wird gestoppt, und der aktuelle Zeitwert wird auf "0" gesetzt.

13.9 ---(SV) Zeit als verlängerten Impuls starten

Symbol

Englisch	Deutsch
<T-Nr.>	<T-Nr.>
---(SE)	**---(SV)**
<Zeitwert>	<Zeitwert>

Parameter	Datentyp	Speicherbereich	Beschreibung
<T-Nr.>	TIMER	T	Nummer der Zeit, Anzahl der Zeiten hängt von der CPU ab
<Zeitwert>	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert

Beschreibung

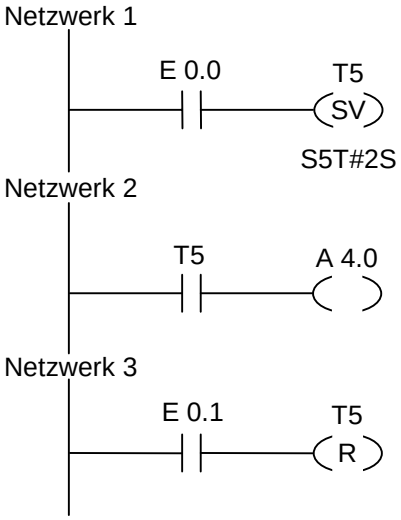
---(SV) (Zeit als verlängerten Impuls starten) startet die angegebene Zeit mit dem **<Zeitwert>**, wenn im VKE eine steigende Flanke auftritt. Die Zeit läuft mit dem angegebenen Zeitintervall weiter, auch wenn das VKE auf "0" geht, bevor die Zeit abgelaufen ist. Eine Signalzustandsabfrage nach "1" ergibt "1", solange die Zeit läuft. Die Zeit wird mit dem angegebenen Zeitwert neu gestartet, wenn das VKE von "0" auf "1" wechselt, während die Zeit läuft.

Siehe auch "Speicherbereiche und Komponenten einer Zeit" und S_VIMP (Zeit als verlängerten Impuls parametrieren und starten).

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Die Zeit läuft mit dem angegebenen Zeitwert weiter, ohne von einer negativen Flanke im VKE beeinträchtigt zu werden. Wechselt der Signalzustand an E 0.0 von "0" auf "1", bevor die Zeit abgelaufen ist, wird die Zeit neu gestartet. Ausgang A 4.0 ist "1", solange die Zeit läuft. Wechselt der Signalzustand an Eingang E 0.1 von "0" auf "1", dann wird die Zeit T5 zurückgesetzt, d.h. sie wird gestoppt, und der aktuelle Zeitwert wird auf "0" gesetzt.

13.10 ---(SE) Zeit als Einschaltverzögerung starten

Symbol

Englisch	Deutsch
<T-Nr.>	<T-Nr.>
---(SD)	**---(SE)**
<Zeitwert>	<Zeitwert>

Parameter	Datentyp	Speicherbereich	Beschreibung
<T-Nr.>	TIMER	T	Nummer der Zeit, Anzahl der Zeiten hängt von der CPU ab
<Zeitwert>	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert

Beschreibung

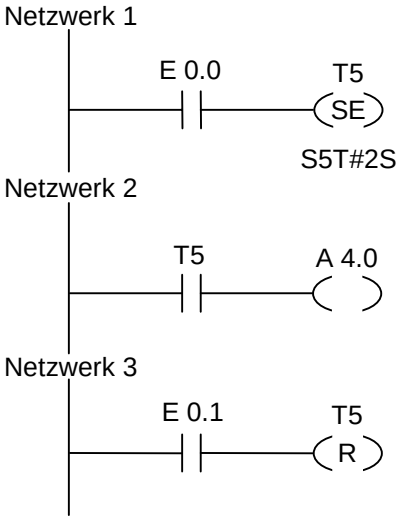
---(SE) (Zeit als Einschaltverzögerung starten) startet die angegebene Zeit mit dem **<Zeitwert>**, wenn im VKE eine steigende Flanke auftritt. Eine Signalzustandsabfrage nach "1" ergibt "1", wenn der **<Zeitwert>** fehlerfrei abgelaufen ist und das VKE noch immer "1" ist. Wechselt das VKE von "1" auf "0", während die Zeit läuft, wird diese zurückgesetzt. In diesem Fall ergibt die Signalzustandsabfrage nach "1" das Ergebnis "0".

Siehe auch "Speicherbereiche und Komponenten einer Zeit" und S_EVERZ (Zeit als Einschaltverzögerung starten).

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Läuft die Zeit ab und der Signalzustand an E 0.0 ist noch immer "1", dann ist Ausgang A 4.0 "1". Wechselt der Signalzustand an E 0.0 von "1" auf "0", während die Zeit läuft, wird diese zurückgesetzt und A 4.0 ist "0". Wechselt der Signalzustand an Eingang E 0.1 von "0" auf "1", dann wird die Zeit T5 zurückgesetzt, d.h. sie wird gestoppt, und der aktuelle Zeitwert wird auf "0" gesetzt.

13.11 ---(SS) Zeit als speichernde Einschaltverzögerung starten

Symbol

Englisch	Deutsch
<T-Nr.>	<T-Nr.>
---(SS)	**---(SS)**
<Zeitwert>	<Zeitwert>

Parameter	Datentyp	Speicherbereich	Beschreibung
<T-Nr.>	TIMER	T	Nummer der Zeit, Anzahl der Zeiten hängt von der CPU ab
<Zeitwert>	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert

Beschreibung

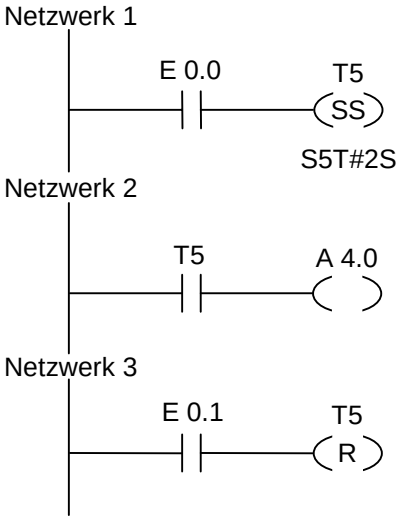
---(SS) (Zeit als speichernde Einschaltverzögerung starten) startet die angegebene Zeit, wenn im VKE eine steigende Flanke auftritt. Der Signalzustand der Zeit ist "1", wenn die Zeit abgelaufen ist. Ein Neustart der Zeit ist erst dann möglich, wenn diese explizit zurückgesetzt wurde. Nur durch ein Rücksetzen kann der Zustand der Zeit auf "0" gesetzt werden.

Siehe auch "Speicherbereiche und Komponenten einer Zeit" und S_SEVERZ (Zeit als speichernde Einschaltverzögerung parametrieren und starten).

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand an Eingang E 0.0 von "0" auf "1" (steigende Flanke im VKE), dann wird die Zeit T5 gestartet. Wechselt der Signalzustand an E 0.0 von "0" auf "1", bevor die Zeit abgelaufen ist, wird die Zeit neu gestartet. Ausgang A 4.0 ist "1", wenn die Zeit abgelaufen ist. Ist der Signalzustand an Eingang E 0.1 "1", dann wird die Zeit T5 zurückgesetzt, d.h., sie wird gestoppt, und der aktuelle Zeitwert wird auf "0" gesetzt.

13.12 ---(SA) Zeit als Ausschaltverzögerung starten

Symbol

Englisch	**Deutsch**
<T-Nr.>	<T-Nr.>
---(SF)	**---(SA)**
<Zeitwert>	<Zeitwert>

Parameter	**Datentyp**	**Speicherbereich**	**Beschreibung**
<T-Nr.>	TIMER	T	Nummer der Zeit, Anzahl der Zeiten hängt von der CPU ab
<Zeitwert>	S5TIME	E, A, M, L, D	Voreingestellter Zeitwert

Beschreibung

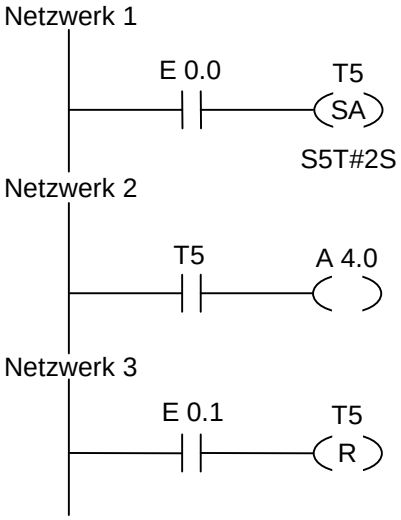
---(SA) (Zeit als Ausschaltverzögerung) startet die angegebene Zeit bei einer fallenden Flanke im VKE. Die Signalzustandsabfrage nach "1" ergibt "1", wenn das VKE "1" ist bzw. solange die Zeit mit dem **<Zeitwert>** läuft. Die Zeit wird zurückgesetzt, wenn das VKE von "0" auf "1" wechselt, während die Zeit läuft. Die Zeit wird immer dann neu gestartet, wenn das VKE von "1" auf "0" wechselt.

Siehe auch "Speicherbereiche und Komponenten einer Zeit" und S_AVERZ (Zeit als Ausschaltverzögerung starten).

Statuswort

	BIE	**A1**	**A0**	**OV**	**OS**	**OR**	**STA**	**VKE**	**/ER**
schreibt:	-	-	-	-	-	0	-	-	0

Beispiel



Wechselt der Signalzustand an Eingang E 0.0 von "1" auf "0", dann wird die Zeit gestartet.

A 4.0 ist "1", wenn E 0.0 "1" ist oder die Zeit läuft. Wechselt der Signalzustand an Eingang E 0.1 von "0" auf "1", dann wird die Zeit T5 zurückgesetzt, d.h. sie wird gestoppt, und der aktuelle Zeitwert wird auf "0" gesetzt.

14 Wortverknüpfung

14.1 Wortverknüpfungsoperationen Übersicht

Beschreibung

Wortverknüpfungsoperationen verknüpfen Paare von Wörtern und Doppelwörtern bitweise entsprechend der Booleschen Logik. Sie werden jeweils durch den Signalzustand "1" am Freigabeeingang EN aktiviert.

Ist das Ergebnis an Ausgang OUT ungleich "0", wird Bit A1 des Statusworts auf "1" gesetzt.
Ist das Ergebnis an Ausgang OUT gleich "0", wird Bit A1 des Statusworts auf "0" gesetzt.

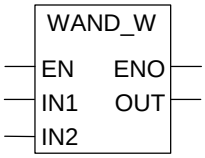
Folgende Operationen stehen Ihnen für Wortverknüpfungen zur Verfügung:

- WAND_W 16 Bit UND verknüpfen
- WOR_W 16 Bit ODER verknüpfen
- WXOR_W 16 Bit EXKLUSIV ODER verknüpfen

- WAND_DW 32 Bit UND verknüpfen
- WOR_DW 32 Bit ODER verknüpfen
- WXOR_DW 32 Bit EXKLUSIV ODER verknüpfen

14.2 WAND_W 16 Bit UND verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	WORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	WORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	WORD	E, A, M, L, D	Ergebniswort der Verknüpfung

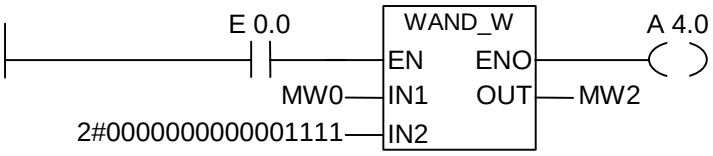
Beschreibung

WAND_W (16 Bit UND verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch UND. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist. Nur die Bits 0 bis 3 von MW0 sind relevant, die restlichen Bits werden vom Bitmuster des Worts in IN2 maskiert:

MW0 = 01010101 01010101

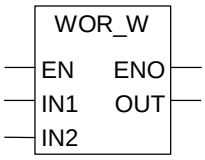
IN2 = 00000000 00001111

MW0 UND IN2 = MW2 = 00000000 00000101

A 4.0 ist "1", wenn die Operation ausgeführt wird.

14.3 WOR_W 16 Bit ODER verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	WORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	WORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	WORD	E, A, M, L, D	Ergebniswort der Verknüpfung

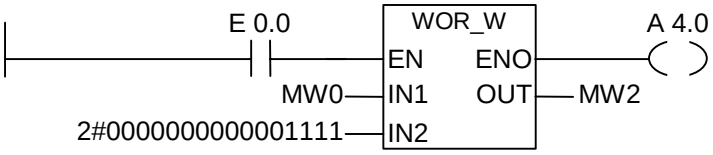
Beschreibung

WOR_W (16 Bit ODER verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch ODER. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist. Bits 0 bis 3 werden auf "1" gesetzt, alle anderen Bits von MW0 werden nicht verändert.

MW0 = 01010101 01010101

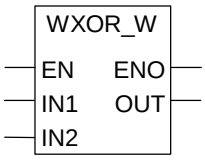
IN2 = 00000000 00001111

MW0 ODER IN2 = MW2 = 01010101 01011111

A 4.0 ist "1", wenn die Operation ausgeführt wird.

14.4 WXOR_W 16 Bit Exklusiv ODER verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	WORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	WORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	WORD	E, A, M, L, D	Ergebniswort der Verknüpfung

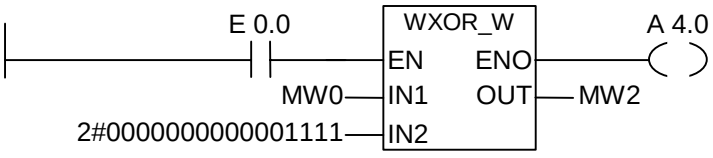
Beschreibung

WXOR_W (16 Bit Exklusiv ODER verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch Exklusiv ODER. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist:

MW0 = 01010101 01010101

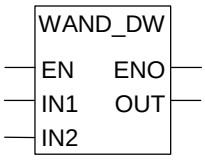
IN2 = 00000000 00001111

MW0 XOR IN2 = MW2 = 01010101 01011010

A 4.0 ist "1", wenn die Operation ausgeführt wird.

14.5 WAND_DW 32 Bit UND verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DWORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	DWORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Verknüpfung

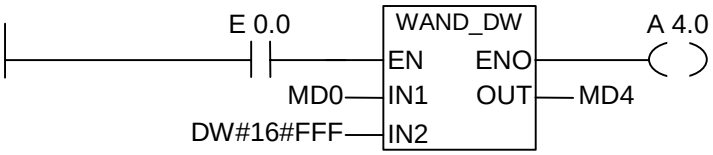
Beschreibung

WAND_DW (32 Bit UND verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch UND. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist. Nur die Bits 0 und 11 von MD0 sind relevant, die restlichen Bits werden vom Bitmuster von IN2 maskiert:

MD0 = 01010101 01010101 01010101 01010101

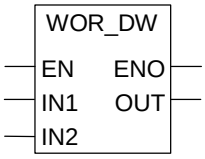
IN2 = 00000000 00000000 00001111 11111111

MD0 UND IN2 = MD4 = 00000000 00000000 00000101 01010101

A 4.0 ist "1", wenn die Operation ausgeführt wird.

14.6 WOR_DW 32 Bit ODER verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DWORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	DWORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Verknüpfung

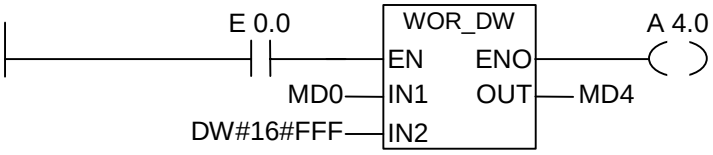
Beschreibung

WOR_DW (32 Bit ODER verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch ODER. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist. Die Bits 0 bis 11 werden auf "1" gesetzt. Die restlichen Bits von MD0 werden nicht verändert:

MD0 = 01010101 01010101 01010101 01010101

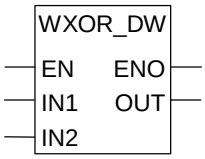
IN2 = 00000000 00000000 00001111 11111111

MD0 ODER IN2 = MD4 = 01010101 01010101 01011111 11111111

A 4.0 ist "1", wenn die Operation ausgeführt wird.

14.7 WXOR_DW 32 Bit Exklusiv ODER verknüpfen

Symbol



Parameter	Datentyp	Speicherbereich	Beschreibung
EN	BOOL	E, A, M, L, D	Freigabeeingang
ENO	BOOL	E, A, M, L, D	Freigabeausgang
IN1	DWORD	E, A, M, L, D	Erster Wert der Verknüpfung
IN2	DWORD	E, A, M, L, D	Zweiter Wert der Verknüpfung
OUT	DWORD	E, A, M, L, D	Ergebnisdoppelwort der Verknüpfung

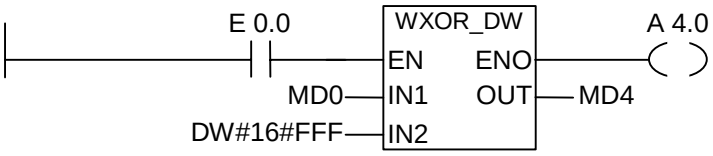
Beschreibung

WXOR_DW (32 Bit Exklusiv ODER verknüpfen) wird durch den Signalzustand "1" am Freigabeeingang (EN) aktiviert und verknüpft die beiden Wortwerte von IN1 und IN2 bitweise durch Exklusiv ODER. Die Werte werden als reine Bitmuster ausgewertet. Das Ergebnis kann an Ausgang OUT abgefragt werden. ENO hat den gleichen Signalzustand wie EN.

Statuswort

	BIE	A1	A0	OV	OS	OR	STA	VKE	/ER
schreibt:	1	X	0	0	-	X	1	1	1

Beispiel



Die Operation wird ausgeführt, wenn E 0.0 = 1 ist:

MD0 = 01010101 01010101 01010101 01010101

IN2 = 00000000 00000000 00001111 11111111

MD4 = MD0 XOR IN2 = 01010101 01010101 01011010 10101010

A 4.0 ist "1", wenn die Operation ausgeführt wird.

A KOP-Operationen Übersicht

A.1 KOP-Operationen sortiert nach deutscher Mnemonik (SIMATIC)

Deutsche Mnemonik	Englische Mnemonik	Operation/Funktion	Beschreibung
---| |---	---| |---	Bitverknüpfung	Schließerkontakt
---|/|---	---|/|---	Bitverknüpfung	Öffnerkontakt
---( )	---( )	Bitverknüpfung	Relaisspule, Ausgang
---(#)---	---(#)---	Bitverknüpfung	Konnektor
==0 ---| |---	==0 ---| |---	Statusbits	Ergebnisbit bei gleich 0
<>0 ---| |---	<>0 ---| |---	Statusbits	Ergebnisbit bei ungleich 0
>0 ---| |---	>0 ---| |---	Statusbits	Ergebnisbit bei größer 0
<0 ---| |---	<0 ---| |---	Statusbits	Ergebnisbit bei kleiner 0
>=0 ---| |---	>=0 ---| |---	Statusbits	Ergebnisbit bei größer gleich 0
<=0 ---| |---	<=0 ---| |---	Statusbits	Ergebnisbit bei kleiner gleich 0
ABS	ABS	Gleitpunkt-Funktion	Bilden des Absolutwertes einer Gleitpunktzahl
ACOS	ACOS	Gleitpunkt-Funktion	Bilden des Arcuscosinuswerts
ADD_DI	ADD_DI	Festpunkt-Funktion	Ganze Zahlen addieren (32 Bit)
ADD_I	ADD_I	Festpunkt-Funktion	Ganze Zahlen addieren (16 Bit)
ADD_R	ADD_R	Gleitpunkt-Funktion	Gleitpunktzahlen addieren
ASIN	ASIN	Gleitpunkt-Funktion	Bilden des Arcussinuswerts
ATAN	ATAN	Gleitpunkt-Funktion	Bilden des Arcustangenswerts
BCD_DI	BCD_DI	Umwandler	BCD-Zahl in 32-Bit-Ganzzahl wandeln
BCD_I	BCD_I	Umwandler	BCD-Zahl in 16-Bit-Ganzzahl wandeln
BIE ---| |---	BR ---| |---	Statusbits	Störungsbit BIE-Register
----(CALL)	----(CALL)	Programmsteuerung	FC/SFC aufrufen ohne Parameter
CALL_FB	CALL_FB	Programmsteuerung	FB als Box aufrufen
CALL_FC	CALL_FC	Programmsteuerung	FC als Box aufrufen
CALL_SFB	CALL_SFB	Programmsteuerung	System-FB als Box aufrufen
CALL_SFC	CALL_SFC	Programmsteuerung	System-FC als Box aufrufen
CEIL	CEIL	Umwandler	Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen
CMP ? D	CMP ? D	Vergleicher	Ganze Zahlen vergleichen (32 Bit)
CMP ? I	CMP ? I	Vergleicher	Ganze Zahlen vergleichen (16 Bit)
CMP ? R	CMP ? R	Vergleicher	Gleitpunktzahlen vergleichen
COS	COS	Gleitpunkt-Funktion	Bildes des Cosinuswerts
DI_BCD	DI_BCD	Umwandler	32-Bit-Ganzzahl in BCD-Zahl wandeln
DI_R	DI_R	Umwandler	32-Bit-Ganzzahl in Gleitpunktzahl wandeln

Deutsche Mnemonik	Englische Mnemonik	Operation/Funktion	Beschreibung
DIV_DI	DIV_DI	Festpunkt-Funktion	Ganze Zahlen dividieren (32 Bit)
DIV_I	DIV_I	Festpunkt-Funktion	Ganze Zahlen dividieren (16 Bit)
DIV_R	DIV_R	Gleitpunkt-Funktion	Gleitpunktzahlen dividieren
EXP	EXP	Gleitpunkt-Funktion	Bilden des Exponentialwerts
FLOOR	FLOOR	Umwandler	Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen
I_BCD	I_BCD	Umwandler	16-Bit-Ganzzahl in BCD-Zahl wandeln
I_DI	I_DI	Umwandler	16-Bit-Ganzzahl in 32-Bit-Ganzzahl wandeln
INV_I	INV_I	Umwandler	1er Komplement zu 16-Bit-Ganzzahl erzeugen
INV_DI	INV_DI	Umwandler	1er Komplement zu 32-Bit-Ganzzahl erzeugen
---(JMP)	---(JMP)	Sprünge	Springe im Baustein wenn 1
---(JMPN)	---(JMPN)	Sprünge	Springe im Baustein wenn 0
LABEL	LABEL	Sprünge	Sprungmarke
LN	LN	Gleitpunkt-Funktion	Bilden des natürlichen Logarithmus
---(MCR<)	---(MCR<)	Programmsteuerung	Master Control Relay einschalten
---(MCR>)	---(MCR>)	Programmsteuerung	Master Control Relay ausschalten
---(MCRA)	---(MCRA)	Programmsteuerung	Master Control Relay Anfang
---(MCRD)	---(MCRD)	Programmsteuerung	Master Control Relay Ende
MOD_DI	MOD_DI	Festpunkt-Funktion	Divisionsrest gewinnen (32 Bit)
MOVE	MOVE	Verschieben	Wert übertragen
MUL_DI	MUL_DI	Festpunkt-Funktion	Ganze Zahlen multiplizieren (32 Bit)
MUL_I	MUL_I	Festpunkt-Funktion	Ganze Zahlen multiplizieren (16 Bit)
MUL_R	MUL_R	Gleitpunkt-Funktion	Gleitpunktzahlen multiplizieren
---(N)---	---(N)---	Bitverknüpfung	Flanke 1 -> 0 abfragen
NEG	NEG	Bitverknüpfung	Signalflanke 1 -> 0 abfragen
NEG_DI	NEG_DI	Bitverknüpfung	2er Komplement zu 32-Bit-Ganzzahl erzeugen
NEG_I	NEG_I	Umwandler	2er Komplement zu 16-Bit-Ganzzahl erzeugen
NEG_R	NEG_R	Umwandler	Vorzeichen einer Gleitpunktzahl wechseln
---| NOT |---	---| NOT |---	Bitverknüpfung	Verknüpfungergebnis invertieren
---(OPN)	---(OPN)	DB-Aufruf	Datenbaustein öffnen
OS ---| |---	OS ---| |---	Statusbits	Störungsbit Überlauf gespeichert
OV ---| |---	OV ---| |---	Statusbits	Störungsbit Überlauf
---(P)---	---(P)---	Bitverknüpfung	Flanke 0 -> 1 abfragen
POS	POS	Bitverknüpfung	Signalflanke 0 -> 1 abfragen
---(R)	---(R)	Bitverknüpfung	Ausgang rücksetzen
---(RET)	---(RET)	Programmsteuerung	Springe zurück
ROL_DW	ROL_DW	Schieben/Rotieren	32 Bit links rotieren
ROR_DW	ROR_DW	Umwandler	32 Bit rechts rotieren
ROUND	ROUND	Schieben/Rotieren	Zahl runden
RS	RS	Bitverknüpfung	Flipflop rücksetzen setzen
---(S)	---(S)	Bitverknüpfung	Ausgang setzen
---(SA)	---(SF)	Zeiten	Zeit als Ausschaltverzögerung starten
---(SAVE)	---(SAVE)	Bitverknüpfung	Verknüpfungsergebnis in BIE-Register laden
S_AVERZ	S_OFFDT	Zeiten	Zeit als Ausschaltverzögerung parametrieren und starten

Deutsche Mnemonik	Englische Mnemonik	Operation/Funktion	Beschreibung
---(SE)	---(SD)	Zeiten	Zeit als Einschaltverzögerung starten
S_EVERZ	S_ODT	Zeiten	Zeit als Einschaltverzögerung parametrieren und starten
SHL_DW	SHL_DW	Schieben/Rotieren	32 Bit links schieben
SHL_W	SHL_W	Schieben/Rotieren	16 Bit links schieben
SHR_DI	SHR_DI	Schieben/Rotieren	32-Bit-Ganzzahl rechts schieben
SHR_DW	SHR_DW	Schieben/Rotieren	32 Bit rechts schieben
SHR_I	SHR_I	Schieben/Rotieren	16-Bit-Ganzzahl rechts schieben
SHR_W	SHR_W	Schieben/Rotieren	16 Bit rechts schieben
---(SI)	---(SP)	Zeiten	Zeit als Impuls starten
S_IMPULS	S_PULSE	Zeiten	Zeit als Impuls parametrieren und starten
SIN	SIN	Gleitpunkt-Funktion	Bilden des Sinuswerts
SQR	SQR	Gleitpunkt-Funktion	Bilden des Quadrats
SQRT	SQRT	Gleitpunkt-Funktion	Bilden der Quadratwurzel
SR	SR	Bitverknüpfung	Flipflop setzen rücksetzen
---(SS)	---(SS)	Zeiten	Zeit als speichernde Einschaltverzögerung starten
S_SEVERZ	S_ODTS	Zeiten	Zeit als speichernde Einschaltverzögerung parametrieren und starten
SUB_DI	SUB_DI	Festpunkt-Funktion	Ganze Zahlen subtrahieren (32 Bit)
SUB_I	SUB_I	Festpunkt-Funktion	Ganze Zahlen subtrahieren (16 Bit)
SUB_R	SUB_R	Gleitpunkt-Funktion	Gleitpunktzahlen subtrahieren
---(SV)	---(SE)	Zeiten	Zeit als verlängerten Impuls starten
S_VIMP	S_PEXT	Zeiten	Zeit als verlängerten Impuls parametrieren und starten
---(SZ)	---(SC)	Zähler	Zähleranfangswert setzen
TAN	TAN	Gleitpunkt-Funktion	Bilden des Tangenswerts
TRUNC	TRUNC	Umwandler	Ganze Zahl erzeugen
UO ---| |---	UO ---| |---	Statusbits	Störungsbit Ungültige Operation
WAND_DW	WAND_DW	Wortverknüpfung	32 Bit UND verknüpfen
WAND_W	WAND_W	Wortverknüpfung	16 Bit UND verknüpfen
WOR_DW	WOR_DW	Wortverknüpfung	32 Bit ODER verknüpfen
WOR_W	WOR_W	Wortverknüpfung	16 Bit ODER verknüpfen
WXOR_DW	WXOR_DW	Wortverknüpfung	32 Bit Exklusiv ODER verknüpfen
WXOR_W	WXOR_W	Wortverknüpfung	16 Bit Exklusiv ODER verknüpfen
ZAEHLER	S_CUD	Zähler	Parametrieren und vorwärts-/rückwärtszählen
----(ZR)	----(CD)	Zähler	Rückwärtszählen
Z_RUECK	----(S_CD)	Zähler	Parametrieren und rückwärtszählen
---(ZV)	----(CU)	Zähler	Vorwärtszählen
Z_VORW	S_CU	Zähler	Parametrieren und vorwärtszählen

A.2 KOP-Operationen sortiert nach englischer Mnemonik (International)

Englische Mnemonik	Deutsche Mnemonik	Operation/Funktion	Beschreibung
---|/|---	---|/|---	Bitverknüpfung	Öffnerkontakt
---| |---	---| |---	Bitverknüpfung	Schließerkontakt
---( )	---( )	Bitverknüpfung	Relaisspule, Ausgang
---(#)---	---(#)---	Bitverknüpfung	Konnektor
==0 ---| |---	==0 ---| |---	Statusbits	Ergebnisbit bei gleich 0
<>0 ---| |---	<>0 ---| |---	Statusbits	Ergebnisbit bei ungleich 0
>0 ---| |---	>0 ---| |---	Statusbits	Ergebnisbit bei größer 0
<0 ---| |---	<0 ---| |---	Statusbits	Ergebnisbit bei kleiner 0
>=0 ---| |---	>=0 ---| |---	Statusbits	Ergebnisbit bei größer gleich 0
<=0 ---| |---	<=0 ---| |---	Statusbits	Ergebnisbit bei kleiner gleich 0
ABS	ABS	Gleitpunkt-Funktion	Bilden des Absolutwertes einer Gleitpunktzahl
ACOS	ACOS	Gleitpunkt-Funktion	Bilden des Arcuscosinuswerts
ADD_DI	ADD_DI	Festpunkt-Funktion	Ganze Zahlen addieren (32 Bit)
ADD_I	ADD_I	Festpunkt-Funktion	Ganze Zahlen addieren (16 Bit)
ADD_R	ADD_R	Gleitpunkt-Funktion	Gleitpunktzahlen addieren
ASIN	ASIN	Gleitpunkt-Funktion	Bilden des Arcussinuswerts
ATAN	ATAN	Gleitpunkt-Funktion	Bilden des Arcustangenswerts
BCD_DI	BCD_DI	Umwandler	BCD-Zahl in 32-Bit-Ganzzahl wandeln
BCD_I	BCD_I	Umwandler	BCD-Zahl in 16-Bit-Ganzzahl wandeln
BR ---| |---	BIE ---| |---	Statusbits	Störungsbit BIE-Register
----(CALL)	----(CALL)	Programmsteuerung	FC/SFC aufrufen ohne Parameter
CALL_FB	CALL_FB	Programmsteuerung	FB als Box aufrufen
CALL_FC	CALL_FC	Programmsteuerung	FC als Box aufrufen
CALL_SFB	CALL_SFB	Programmsteuerung	System-FB als Box aufrufen
CALL_SFC	CALL_SFC	Programmsteuerung	System-FC als Box aufrufen
----(CD)	----(ZR)	Zähler	Rückwärtszählen
CEIL	CEIL	Umwandler	Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen
CMP ? D	CMP ? D	Vergleicher	Ganze Zahlen vergleichen (32 Bit)
CMP ? I	CMP ? I	Vergleicher	Ganze Zahlen vergleichen (16 Bit)
CMP ? R	CMP ? R	Vergleicher	Gleitpunktzahlen vergleichen
COS	COS	Gleitpunkt-Funktion	Bildes des Cosinuswerts
----(CU)	---(ZV)	Zähler	Vorwärtszählen
DI_BCD	DI_BCD	Umwandler	32-Bit-Ganzzahl in BCD-Zahl wandeln
DI_R	DI_R	Festpunkt-Funktion	32-Bit-Ganzzahl in Gleitpunktzahl wandeln
DIV_DI	DIV_DI	Festpunkt-Funktion	Ganze Zahlen dividieren (32 Bit)
DIV_I	DIV_I	Festpunkt-Funktion	Ganze Zahlen dividieren (16 Bit)
DIV_R	DIV_R	Gleitpunkt-Funktion	Gleitpunktzahlen dividieren

Englische Mnemonik	Deutsche Mnemonik	Operation/Funktion	Beschreibung
EXP	EXP	Gleitpunkt-Funktion	Bilden des Exponentialwerts
FLOOR	FLOOR	Umwandler	Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen
I_BCD	I_BCD	Umwandler	16-Bit-Ganzzahl in BCD-Zahl wandeln
I_DI	I_DI	Umwandler	16-Bit-Ganzzahl in 32-Bit-Ganzzahl wandeln
INV_I	INV_I	Umwandler	1er Komplement zu 16-Bit-Ganzzahl erzeugen
INV_DI	INV_DI	Umwandler	1er Komplement zu 32-Bit-Ganzzahl erzeugen
---(JMP)	---(JMP)	Sprünge	Springe im Baustein wenn 1
---(JMPN)	---(JMPN)	Sprünge	Springe im Baustein wenn 0
LABEL	LABEL	Sprünge	Sprungmarke
LN	LN	Gleitpunkt-Funktion	Bilden des natürlichen Logarithmus
---(MCR<)	---(MCR<)	Programmsteuerung	Master Control Relay einschalten
---(MCR>)	---(MCR>)	Programmsteuerung	Master Control Relay ausschalten
---(MCRA)	---(MCRA)	Programmsteuerung	Master Control Relay Anfang
---(MCRD)	---(MCRD)	Programmsteuerung	Master Control Relay Ende
MOD_DI	MOD_DI	Festpunkt-Funktion	Divisionsrest gewinnen (32 Bit)
MOVE	MOVE	Verschieben	Wert übertragen
MUL_DI	MUL_DI	Festpunkt-Funktion	Ganze Zahlen multiplizieren (32 Bit)
MUL_I	MUL_I	Festpunkt-Funktion	Ganze Zahlen multiplizieren (16 Bit)
MUL_R	MUL_R	Gleitpunkt-Funktion	Gleitpunktzahlen multiplizieren
---(N)---	---(N)---	Bitverknüpfung	Flanke 1 -> 0 abfragen
NEG	NEG	Bitverknüpfung	Signalflanke 1 -> 0 abfragen
NEG_DI	NEG_DI	Umwandler	2er Komplement zu 32-Bit-Ganzzahl erzeugen
NEG_I	NEG_I	Umwandler	2er Komplement zu 16-Bit-Ganzzahl erzeugen
NEG_R	NEG_R	Umwandler	Vorzeichen einer Gleitpunktzahl wechseln
---| NOT |---	---| NOT |---	Bitverknüpfung	Verknüpfungsergebnis invertieren
---(OPN)	---(OPN)	DB-Aufruf	Datenbaustein öffnen
OS ---| |---	OS ---| |---	Statusbits	Störungsbit Überlauf gespeichert
OV ---| |---	OV ---| |---	Statusbits	Störungsbit Überlauf
---(P)---	---(P)---	Bitverknüpfung	Flanke 0 -> 1 abfragen
POS	POS	Bitverknüpfung	Signalflanke 0 -> 1 abfragen
---(R)	---(R)	Bitverknüpfung	Ausgang rücksetzen
---(RET)	---(RET)	Programmsteuerung	Springe zurück
ROL_DW	ROL_DW	Schieben/Rotieren	32 Bit links rotieren
ROR_DW	ROR_DW	Schieben/Rotieren	32 Bit rechts rotieren
ROUND	ROUND	Umwandler	Zahl runden
RS	RS	Bitverknüpfung	Flipflop rücksetzen setzen
---(S)	---(S)	Bitverknüpfung	Ausgang setzen
---(SAVE)	---(SAVE)	Bitverknüpfung	Verknüpfungsergebnis in BIE-Register laden
---(SC)	---(SZ)	Zähler	Zähleranfangswert setzen
----(S_CD)	Z_RUECK	Zähler	Parametrieren und rückwärtszählen
S_CU	Z_VORW	Zähler	Parametrieren und vorwärtszählen

Englische Mnemonik	Deutsche Mnemonik	Operation/Funktion	Beschreibung
S_CUD	ZAEHLER	Zähler	Parametrieren und vorwärts-/rückwärtszählen
---(SD)	---(SE)	Zeiten	Zeit als Einschaltverzögerung starten
---(SE)	---(SV)	Zeiten	Zeit als verlängerten Impuls starten
---(SF)	---(SA)	Zeiten	Zeit als Ausschaltverzögerung starten
SHL_DW	SHL_DW	Schieben/Rotieren	32 Bit links schieben
SHL_W	SHL_W	Schieben/Rotieren	16 Bit links schieben
SHR_DI	SHR_DI	Schieben/Rotieren	32-Bit-Ganzzahl rechts schieben
SHR_DW	SHR_DW	Schieben/Rotieren	32 Bit rechts schieben
SHR_I	SHR_I	Schieben/Rotieren	16-Bit-Ganzzahl rechts schieben
SHR_W	SHR_W	Schieben/Rotieren	16 Bit rechts schieben
SIN	SIN	Gleitpunkt-Funktion	Bilden des Sinuswerts
S_ODT	S_EVERZ	Zeiten	Zeit als Einschaltverzögerung parametrieren und starten
S_ODTS	S_SEVERZ	Zeiten	Zeit als speichernde Einschaltverzögerung parametrieren und starten
S_OFFDT	S_AVERZ	Zeiten	Zeit als Ausschaltverzögerung parametrieren und starten
S_OFFDT	S_AVERZ	Zeiten	Zeit als Ausschaltverzögerung parametrieren und starten
---(SP)	---(SI)	Zeiten	Zeit als Impuls starten
S_PEXT	S_VIMP	Zeiten	Zeit als verlängerten Impuls parametrieren und starten
S_PULSE	S_IMPULS	Zeiten	Zeit als Impuls parametrieren und starten
SQR	SQR	Gleitpunkt-Funktion	Bilden des Quadrats
SQRT	SQRT	Gleitpunkt-Funktion	Bilden der Quadratwurzel
SR	SR	Bitverknüpfung	Flipflop setzen rücksetzen
---(SS)	---(SS)	Zeiten	Zeit als speichernde Einschaltverzögerung starten
SUB_DI	SUB_DI	Festpunkt-Funktion	Ganze Zahlen subtrahieren (32 Bit)
SUB_I	SUB_I	Festpunkt-Funktion	Ganze Zahlen subtrahieren (16 Bit)
SUB_R	SUB_R	Gleitpunkt-Funktion	Gleitpunktzahlen subtrahieren
TAN	TAN	Gleitpunkt-Funktion	Bilden des Tangenswerts
TRUNC	TRUNC	Umwandler	Ganze Zahl erzeugen
UO ---| |---	UO ---| |---	Statusbits	Störungsbit Ungültige Operation
WAND_DW	WAND_DW	Wortverknüpfung	32 Bit UND verknüpfen
WAND_W	WAND_W	Wortverknüpfung	16 Bit UND verknüpfen
WOR_DW	WOR_DW	Wortverknüpfung	32 Bit ODER verknüpfen
WOR_W	WOR_W	Wortverknüpfung	16 Bit ODER verknüpfen
WXOR_DW	WXOR_DW	Wortverknüpfung	32 Bit Exklusiv ODER verknüpfen
WXOR_W	WXOR_W	Wortverknüpfung	16 Bit Exklusiv ODER verknüpfen

B Programmierbeispiele

B.1 Programmierbeispiele Übersicht

Praktische Anwendungen

Jede KOP-Operation löst eine bestimmte Funktion aus. Durch Kombination der Operationen in einem Programm können Sie eine breite Palette von Automatisierungsaufgaben ausführen. Hier einige Beispiele für praktische Anwendungen:

- Steuern eines Förderbandes durch Bitverknüpfungsoperationen
- Feststellen der Bewegungsrichtung auf einem Förderband durch Bitverknüpfungsoperationen
- Generieren eines Taktimpulses durch Zeitoperationen
- Überwachen des Lagerbereichs durch Zähl- und Vergleichsoperationen
- Berechnungen mit arithmetischen Operationen für Ganzzahlen
- Einstellen der Zeitdauer für das Beheizen eines Ofens

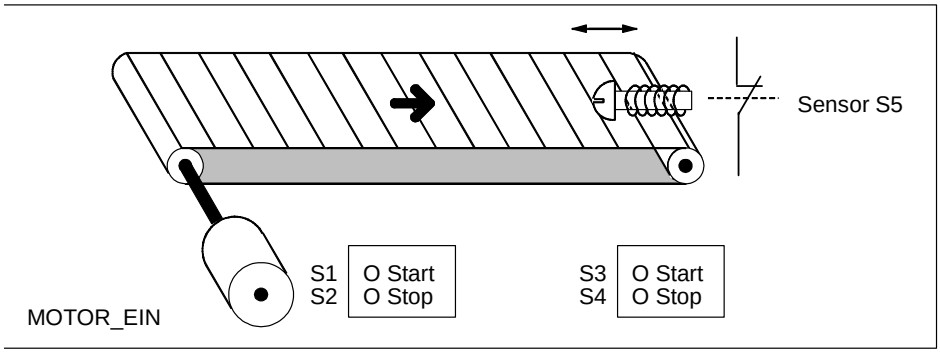
Verwendete Operationen

Mnemonik	Operation	Beschreibung
WAND_W	Wortverknüpfung	16 Bit UND verknüpfen
WOR_W	Wortverknüpfung	16 Bit ODER verknüpfen
Z_RUECK	Zähler	Rückwärtszählen
Z_VORW	Zähler	Vorwärtszählen
---(R)	Bitverknüpfung	Ausgang rücksetzen
---(S)	Bitverknüpfung	Ausgang setzen
---(P)	Bitverknüpfung	Flanke 0 $\rightarrow$ 1 abfragen
ADD_I	Festpunkt-Funktion	Ganze Zahlen addieren (16 Bit)
DIV_I	Festpunkt-Funktion	Ganze Zahlen dividieren (16 Bit)
MUL_I	Festpunkt-Funktion	Ganze Zahlen multiplizieren (16 Bit)
CMP >=I	Vergleicher	Ganze Zahlen vergleichen (16 Bit)
CMP <=I	Vergleicher	Ganze Zahlen vergleichen (16 Bit)
--| |--	Bitverknüpfung	Schließer
--| / |--	Bitverknüpfung	Öffner
--( )	Bitverknüpfung	Relaisspule, Ausgang
---(JMPN)	Sprünge	Springe im Baustein wenn 0 (bedingt)
---(RET)	Programmsteuerung	Springe zurück
MOVE	Verschieben	Wert übertragen
--- (SV)	Zeiten	Zeit als verlängerten Impuls starten

B.2 Bitverknüpfungsoperationen Beispiel

Beispiel 1: Steuern eines Förderbandes

Das folgende Bild zeigt ein Förderband, das elektrisch in Gang gesetzt werden kann. Am Anfang des Bandes befinden sich zwei Druckschalter, S1 für START und S2 für STOP. Am Ende des Bandes befinden sich ebenfalls zwei Druckschalter, S3 für START und S4 für STOP. Das Band kann von beiden Enden aus gestartet oder gestoppt werden. Außerdem stoppt der Sensor S5 das Band, wenn ein Gegenstand auf dem Band dessen Ende erreicht.



Absolute und symbolische Programmierung

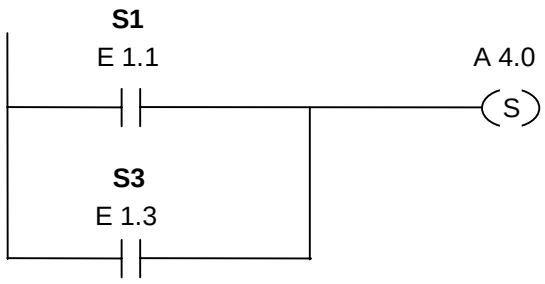
Sie können ein Programm zum Steuern des Förderbandes schreiben, indem Sie die verschiedenen Komponenten des Förderbandsystems mit Hilfe von **absoluten Adressen** oder **Symbolen** darstellen.

Die von Ihnen gewählten Symbole setzen Sie in der Symboltabelle mit den absoluten Adressen in Beziehung (siehe Online-Hilfe zu STEP 7).

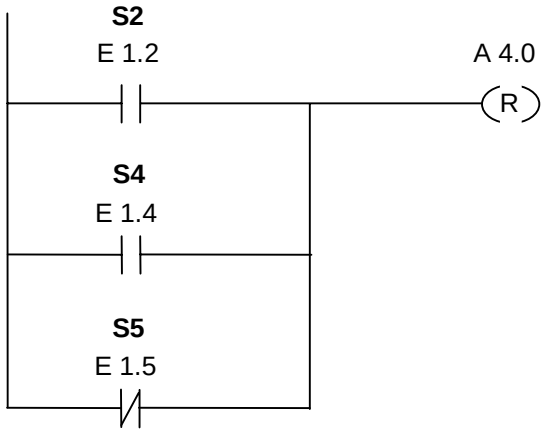
Systemkomponente	Absolute Adresse	Symbol	Symboltabelle
Startschalter	E 1.1	S1	E 1.1 S1
Stoppschalter	E 1.2	S2	E 1.2 S2
Startschalter	E 1.3	S3	E 1.3 S3
Stoppschalter	E 1.4	S4	E 1.4 S4
Sensor	E 1.5	S5	E 1.5 S5
Motor	A 4.0	MOTOR_EIN	A 4.0 MOTOR_EIN

Kontaktplan zum Steuern des Förderbandes

Netzwerk 1: Der Motor wird durch Betätigen eines der beiden Startschalters eingeschaltet.

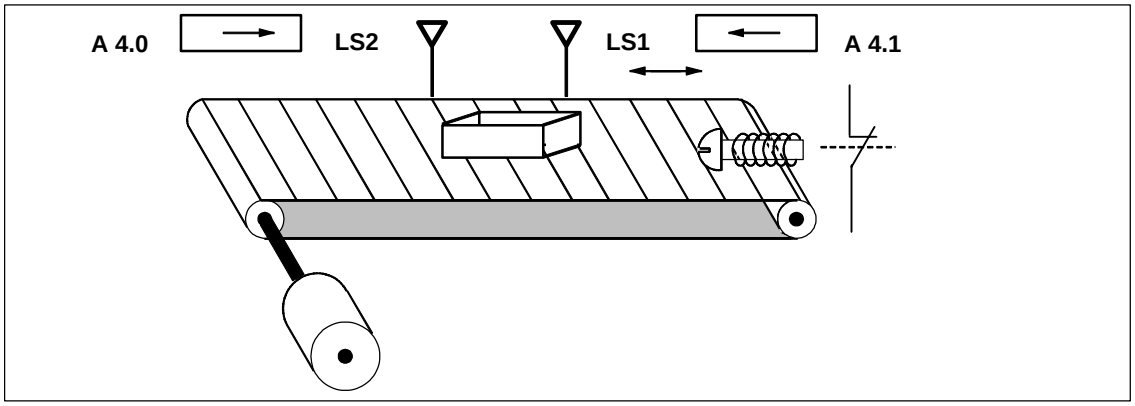


Netzwerk 2: Der Motor wird durch Betätigen eines der beiden Stoppschalter oder durch Ansprechen des Sensors am Ende des Bandes ausgeschaltet.



Beispiel 2: Erfassen der Richtung eines Förderbandes

Das folgende Bild zeigt ein Förderband, das mit zwei Lichtschranken (LS1, LS2) ausgestattet ist. Die Lichtschranken sollen feststellen, in welche Richtung sich ein Paket auf dem Band bewegt.



Absolute und symbolische Programmierung

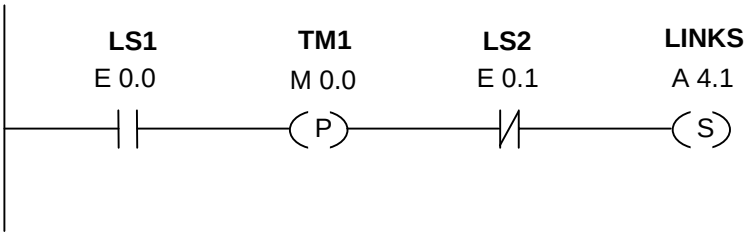
Sie können ein Programm schreiben, das die Richtungsanzeige für das Förderbandsystem aktiviert, indem Sie die verschiedenen Komponenten des Fördersystems mit Hilfe von **absoluten Adressen** oder **Symbolen** darstellen.

Die von Ihnen gewählten Symbole setzen Sie in der Symboltabelle mit den absoluten Adressen in Beziehung (siehe Online-Hilfe zu STEP 7).

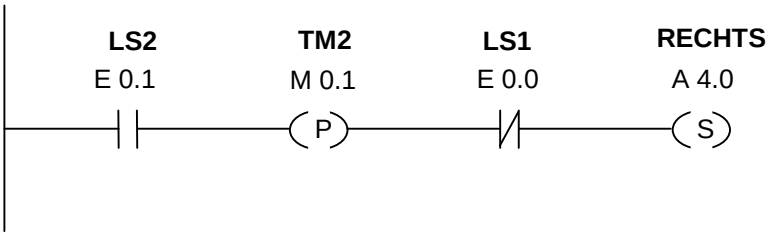
Systemkomponente	Absolute Adresse	Symbol	Symboltabelle
Lichtschranke 1	E 0.0	LS1	E 0.0 LS1
Lichtschranke 2	E 0.1	LS2	E 0.1 LS2
Anzeige für Bewegung nach rechts	A 4.0	RECHTS	A 4.0 RECHTS
Anzeige für Bewegung nach links	A 4.1	LINKS	A 4.1 LINKS
Taktmerker 1	M 0.0	TM1	M 0.0 TM1
Taktmerker 2	M 0.1	TM2	M 0.1 TM2

Kontaktplan zur Richtungserfassung eines Förderbandes

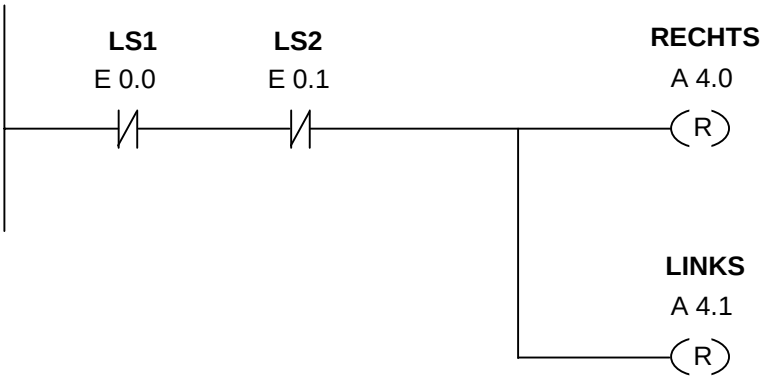
Netzwerk 1: Wenn an E 0.0 ein Wechsel des Signalzustands von "0" auf "1" auftritt (positive Flanke) und gleichzeitig der Signalzustand an E 0.1 "0" ist, dann bewegt sich das Paket auf dem Band nach links.



Netzwerk 2: Wenn an E 0.1 ein Wechsel des Signalzustands von "0" auf "1" auftritt (positive Flanke) und gleichzeitig der Signalzustand an E 0.0 "0" ist, dann bewegt sich das Paket auf dem Band nach rechts.



Netzwerk 3: Ist eine der Lichtschranken unterbrochen, dann befindet sich ein Paket zwischen den Schranken. Die Richtungsanzeiger sind ausgeschaltet.



B.3 Zeitoperationen Beispiel

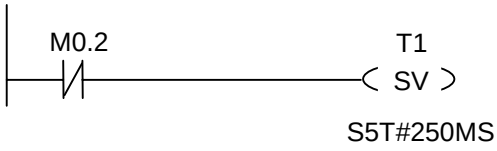
Taktgeber

Zur Erzeugung eines sich periodisch wiederholenden Signals können Sie einen Taktgeber oder ein Blinkrelais verwenden. Taktgeber finden sich häufig in Meldesystemen, die das Blinken von Anzeigeleuchten steuern.

Wenn Sie S7-300 einsetzen, können Sie eine Taktgeberfunktion implementieren, indem Sie die zeitgesteuerte Verarbeitung in speziellen Organisationsbausteinen verwenden.

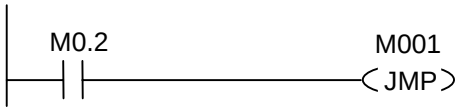
Kontaktplan zum Generieren eines Taktes (Tastverhältnis 1:1)

Netzwerk 1: Wenn der Signalzustand der Zeit T1 "0" ist, dann laden Sie den Zeitwert 250 ms in T1 und starten Sie T1 als verlängerten Impuls.

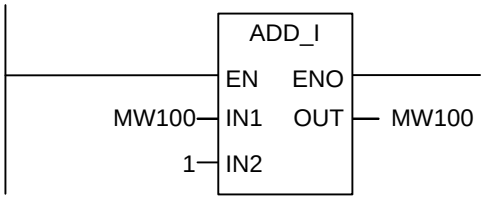


Netzwerk 2: Der Zustand des Timers wird in einem Hilfsmerker zwischengespeichert.

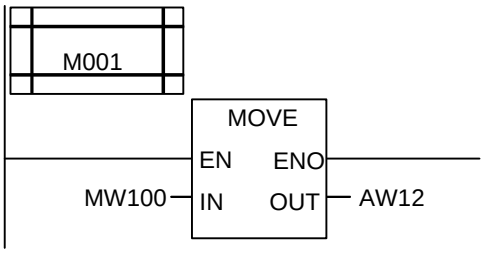
Netzwerk 3: Ist der Signalzustand der Zeit T1 gleich "1", dann springe zur Sprungmarke M001.



Netzwerk 4: Nach jeder abgelaufenen Zeit des Timers T1, wird das Merkerwort 100 um "1" inkrementiert.

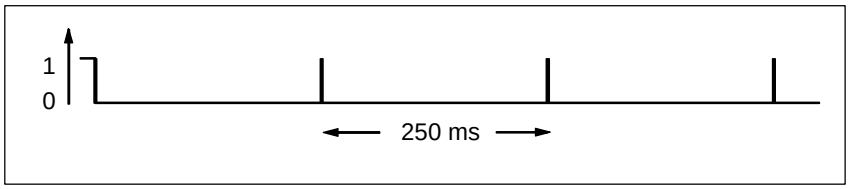


Netzwerk 5: Mit der Operation **MOVE** können Sie sich die unterschiedlichen Taktfrequenzen an den Ausgängen A 12.0 bis A 13.7 anzeigen lassen.

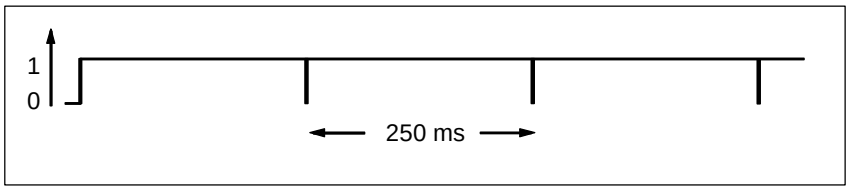


Signalabfrage

Eine Signalabfrage der Zeit T1 liefert für den Öffner M0.2 folgendes Verknüpfungsergebnis:



Sobald die Zeit abgelaufen ist, wird die Zeit erneut gestartet. Daher liefert die Signalabfrage, die von der Anweisung ---I / I--- M0.2 ausgeführt wird, nur kurz den Signalzustand ”1”.
Negiertes VKE-Bit:



Alle 250 ms beträgt das VKE-Bit ”0”. Der Sprung wird ignoriert und der Inhalt des Merkerworts MW100 um ”1” inkrementiert.

Erzielen einer bestimmten Frequenz

Mit den Bits der Merkerbytes MB 101 und MB 100 können Sie folgende Frequenzen erzielen:

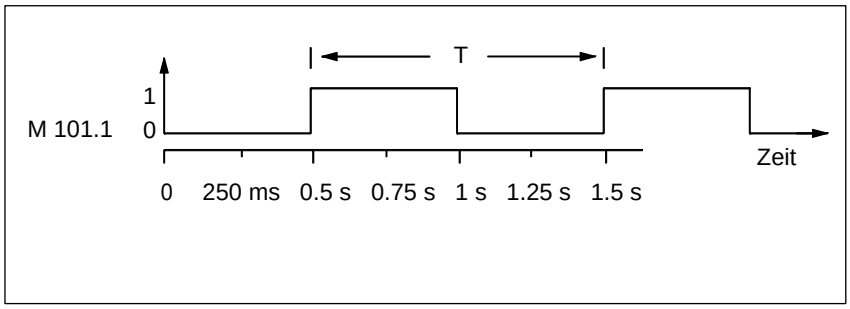
MB 101, MB 100	Frequenz in Hertz	Dauer	
M 101.0	2.0	0.5 s	(250 ms ein / 250 ms aus)
M 101.1	1.0	1 s	(0.5 s ein / 0.5 s aus)
M 101.2	0.5	2 s	(1 s ein / 1 s aus)
M 101.3	0.25	4 s	(2 s ein / 2 s aus)
M 101.4	0.125	8 s	(4 s ein / 4 s aus)
M 101.5	0.0625	16 s	(8 s ein / 8 s aus)
M 101.6	0.03125	32 s	(16 s ein / 16 s aus)
M 101.7	0.015625	64 s	(32 s ein / 32 s aus)
M 100.0	0.0078125	128 s	(64 s ein / 64 s aus)
M 100.1	0.0039062	256 s	(128 s ein / 128 s aus)
M 100.2	0.0019531	512 s	(256 s ein / 256 s aus)
M 100.3	0.0009765	1024 s	(512 s ein / 512 s aus)
M 100.4	0.0004882	2048 s	(1024 s ein / 1024 s aus)
M 100.5	0.0002441	4096 s	(2048 s ein / 2048 s aus)
M 100.6	0.000122	8192 s	(4096 s ein / 4096 s aus)
M 100.7	0.000061	16384 s	(8192 s ein / 8192 s aus)

Signalzustände der Bits von Merkerbyte MB 101

Zyklus	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Zeitwert in ms
0	0	0	0	0	0	0	**0**	0	250
1	0	0	0	0	0	0	**0**	1	250
2	0	0	0	0	0	0	1	0	250
3	0	0	0	0	0	0	1	1	250
4	0	0	0	0	0	1	**0**	0	250
5	0	0	0	0	0	1	**0**	1	250
6	0	0	0	0	0	1	1	0	250
7	0	0	0	0	0	1	1	1	250
8	0	0	0	0	1	0	**0**	0	250
9	0	0	0	0	1	0	**0**	1	250
10	0	0	0	0	1	0	1	0	250
11	0	0	0	0	1	0	1	1	250
12	0	0	0	0	1	1	**0**	0	250

Signalzustand des Merkerbits M 101.1

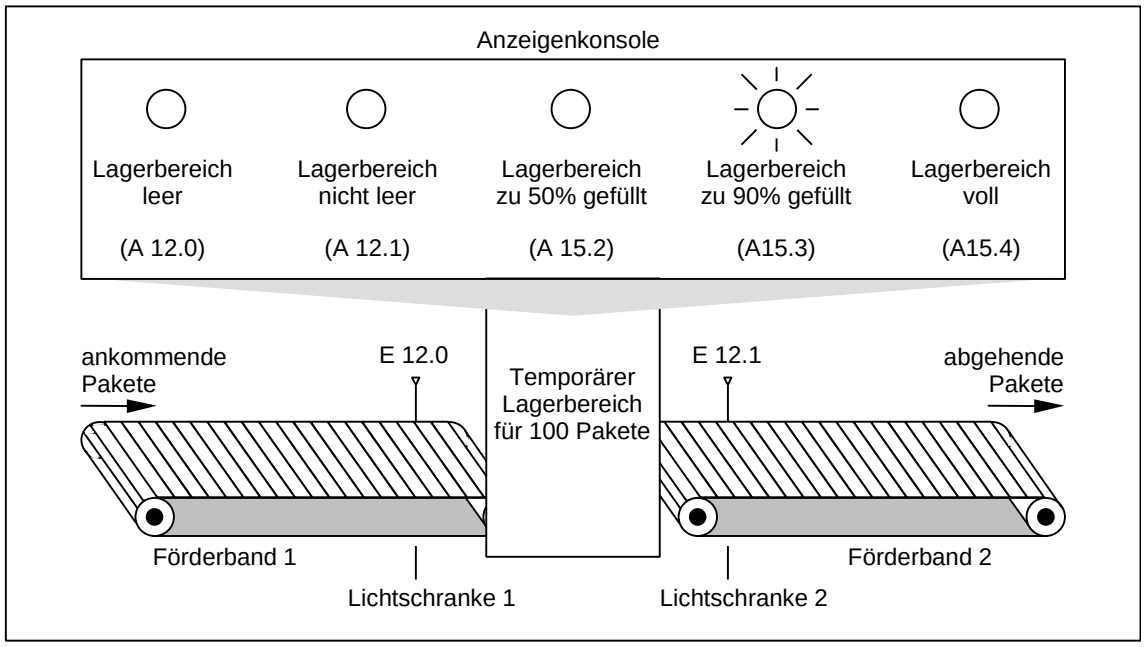
Frequenz = 1/T = 1/1 s = 1 Hz



B.4 Zähl- und Vergleichsoperationen Beispiel

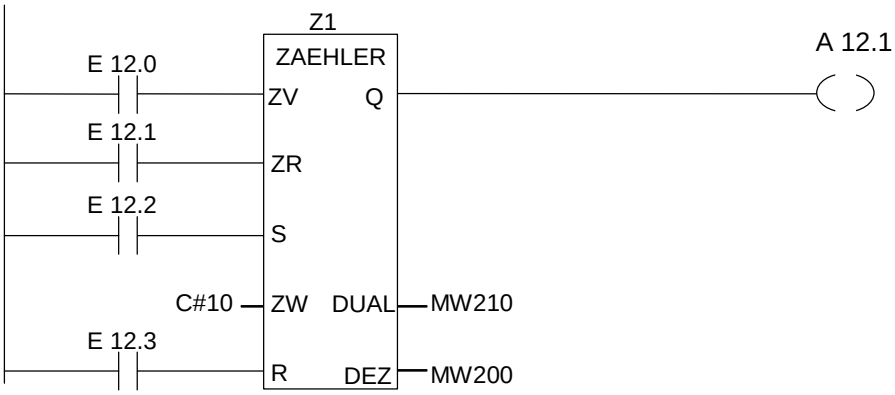
Lagerbereich mit Zähler und Vergleicher

Das folgende Bild zeigt ein System mit zwei Förderbändern und einem temporären Lagerbereich dazwischen. Förderband 1 transportiert die Pakete zum Lagerbereich. Eine Lichtschranke am Ende des Förderbandes 1 neben dem Lagerbereich ermittelt, wie viele Pakete in den Lagerbereich transportiert werden. Förderband 2 transportiert Pakete von diesem temporären Lagerbereich zu einer Laderampe, wo sie zur Auslieferung beim Kunden auf LKW verladen werden. Eine Lichtschranke am Ende des Förderbandes 2 neben dem Lagerbereich ermittelt, wie viele Pakete aus dem Lagerbereich heraus zur Laderampe transportiert werden. Fünf Anzeigeleuchten zeigen an, wie weit der temporäre Lagerbereich gefüllt ist.



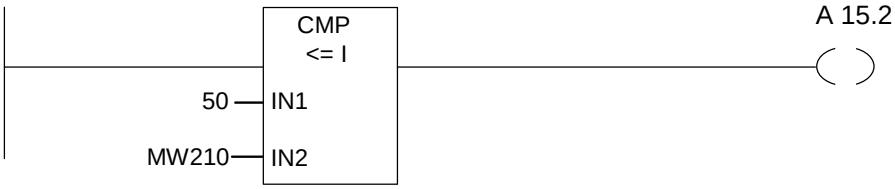
Kontaktplan, der die Anzeigeleuchten aktiviert

Netzwerk 1: Zähler Z1 zählt vorwärts bei einer Signalflanke von "0" auf "1" an Eingang ZV und zählt rückwärts bei einer Signalflanke von "0" auf "1" an Eingang ZR. Mit einer Signalflanke von "0" auf "1" an Eingang S wird der Zählwert auf den Wert von ZW gesetzt. Mit einer Signalflanke von "0" an Eingang R wird der Zählwert auf "0" gesetzt. Im MW200 ist stets der aktuelle Zählwert von Z1 hinterlegt. A12.1 zeigt an "Lagerbereich nicht leer".

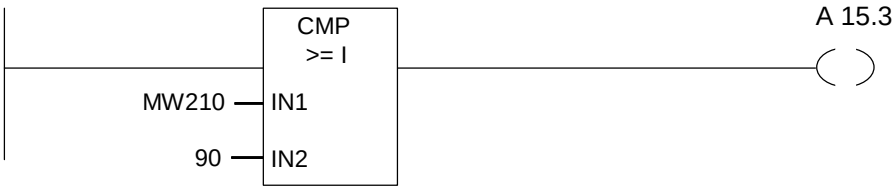


Netzwerk 2: A12.0 zeigt an "Lagerbereich leer".

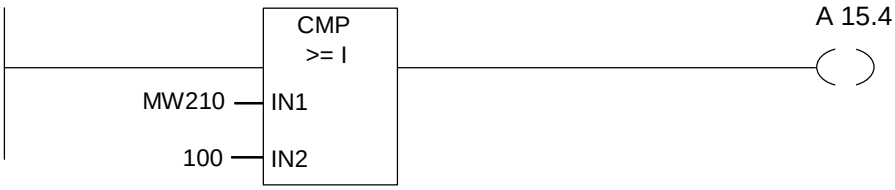
Netzwerk 3: Ist 50 kleiner oder gleich Zählwert (bzw. ist der aktuelle Zählerstand größer oder gleich 50), schaltet sich die Anzeigeleuchte für die Meldung "Lagerbereich zu 50 % voll" ein.



Netzwerk 4: Ist der Zählwert größer oder gleich 90, dann schaltet sich die Anzeigeleuchte für die Meldung "Lagerbereich zu 90 % voll" ein.



Netzwerk 5: Ist der Zählwert größer oder gleich 100, dann schaltet sich die Anzeigeleuchte für die Meldung "Lagerbereich voll" ein.



B.5 Arithmetische Operationen mit Ganzzahlen Beispiel

Berechnen einer Gleichung

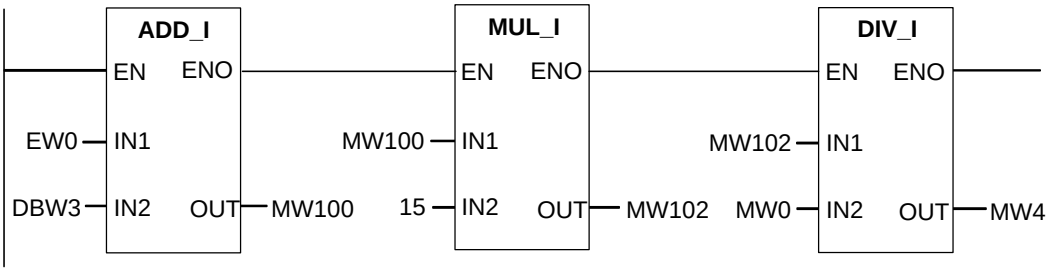
Das folgende Programmbeispiel zeigt, wie Sie mit drei arithmetischen Operationen für Ganzzahlen das gleiche Ergebnis erzielen, wie die folgende Gleichung:

MW4 = ((EW0 + DBW3) x 15) / MW0

Kontaktplan

Netzwerk 1: Datenbaustein DB1 öffnen

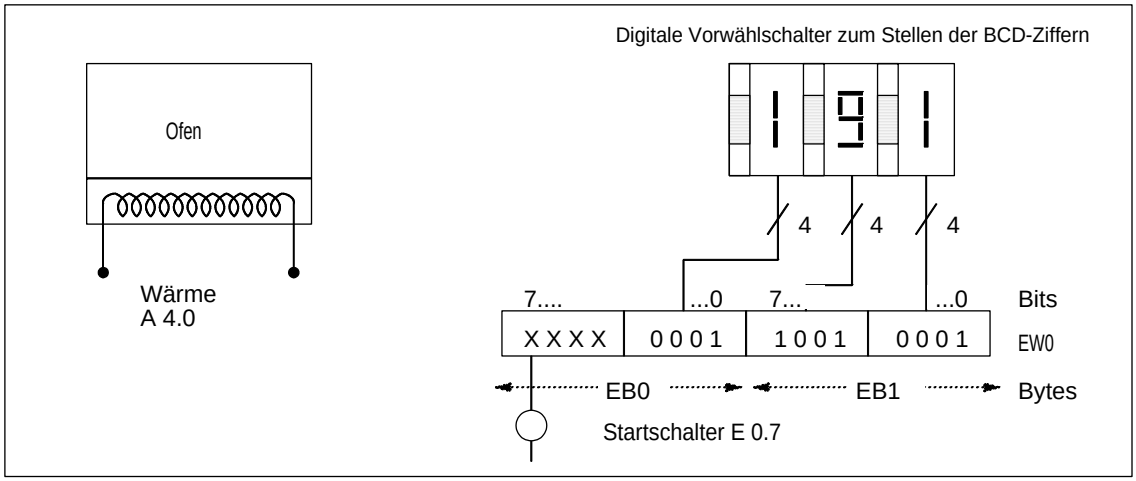
Netzwerk 2: Addiere Eingangswort EW0 und Globaldatenwort DBW3 (der Datenbaustein muß definiert und geöffnet sein). Die Summe wird in MW100 gespeichert. Multipliziere MW100 mit 15. Das Ergebnis wird im Merkerwort MW102 gespeichert. Dividiere MW102 durch MW0.
Das Ergebnis wird in MW4 gespeichert.



B.6 Wortverknüpfungsoperationen Beispiel

Heizen eines Ofens

Der Bediener startet das Heizen des Ofens, indem er den Startschalter drückt. Mit den digitalen Vorwählschaltern kann er die Dauer der Heizzeit festlegen. Der Wert, den er setzt, gibt die Sekunden im binär-codierten Dezimalformat (BCD) an.



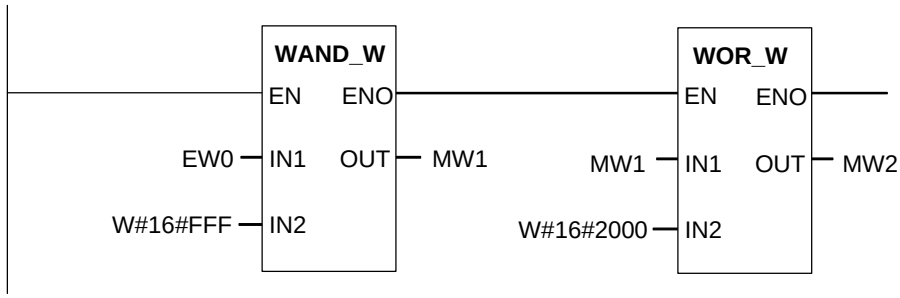
Systemkomponente	Absolute Adresse
Startschalter	E 0.7
Digitale Vorwählschalter für Einer	E 1.0 bis E 1.3
Digitale Vorwählschalter für Zehner	E 1.4 bis E 1.7
Digitale Vorwählschalter für Hunderter	E 0.0 bis E 0.3
Beginn Heizvorgang	A 4.0

Kontaktplan

Netzwerk 1: Wenn die Zeit läuft, dann beginne den Heizvorgang.

Netzwerk 2: Wenn die Zeit läuft, dann beendet die Operation **Springe zurück** die Bearbeitung hier.

Netzwerk 3: Maskiere die Eingangsbits E 0.4 bis E 0.7 (d.h. setze sie auf "0" zurück). Diese Bits der Vorwählschalter-Eingänge werden nicht verwendet. Die 16 Bits der Vorwählschalter-Eingänge werden mit W#16#0FFF nach der Operation **16 Bit UND verknüpfen** verknüpft. Das Ergebnis wird in das Merkerwort MW1 geladen. Um den Zeitwert in Sekunden einzustellen, wird die Voreinstellung mit W#16#2000 nach der Operation **16 Bit ODER verknüpfen** verknüpft. Bit 13 wird auf "1" gesetzt, Bit 12 wird auf "0" rückgesetzt.



Netzwerk 4: Starte die Zeit T1 als verlängerten Impuls, wenn der Schalter gedrückt wird. Merkerwort MW2 wird als Voreinstellung geladen (abgeleitet von der Verknüpfungsoperation von oben).

C Arbeiten mit KOP

C.1 EN-/ENO-Mechanismus

Die Freigabe (EN) und der Freigabeausgang (ENO) der FUP/KOP-Boxen wird mittels des BIE-Bits realisiert.

Wenn EN und ENO beschaltet sind, dann gilt:

ENO = EN AND NOT (Fehler der Box)

Wenn kein Fehler auftritt (Fehler der Box = 1), ist somit ENO = EN.

Der EN-/ENO-Mechanismus wird verwendet für:

- arithmetische Operationen,
- Übertragungs- und Umwandlungsoperationen,
- Schiebe und Rotieroperationen,
- Bausteinaufrufe.

Dieser Mechanismus wird **nicht** verwendet für:

- Vergleicher,
- Zähler,
- Timer.

Um die eigentlichen Befehle der Box werden für den EN-/ENO-Mechanismus zusätzliche AWL-Befehle in Abhängigkeit von vorhandenen Vor- und Nachverknüpfungen generiert. Die vier möglichen Fälle werden am Beispiel eines Addierers gezeigt:

- Addierer mit EN- und mit ENO-Beschaltung
- Addierer mit EN- und ohne ENO-Beschaltung
- Addierer ohne EN- und mit ENO-Beschaltung
- Addierer ohne EN- und ohne ENO-Beschaltung

Hinweis zum Erstellen eigener Bausteine

Wenn Sie Bausteine schreiben wollen, die Sie in FUP/KOP aufrufen wollen, so müssen Sie dafür sorgen, daß beim Verlassen des Bausteins das BIE gesetzt ist. Das vierte Beispiel zeigt, daß dies nicht automatisch der Fall ist. Sie können das BIE nicht als Merker verwenden, da es ständig vom EN/ENO-Mechanismus überschrieben wird. Benutzen Sie statt dessen eine temporäre Variable, in der Sie aufgetretene Fehler speichern. Initialisieren Sie diese Variable (im Beispiel unten: fehler) mit 0. An jeder Stelle im Baustein, an der Sie meinen, daß eine mißlungene Operation einen Fehler für den gesamten Baustein darstellt, setzen Sie unter Zuhilfenahme des EN-/ENO-Mechanismus diese Variable. Dazu reicht ein NOT und eine Setze-Spule. Am Ende des Bausteins programmieren Sie ein Netzwerk:

```
ende: UN fehler
      SAVE
```

Achten Sie darauf, daß dieses Netzwerk in jedem Fall durchlaufen wird, d.h. sie dürfen innerhalb des Bausteins keinen BEB verwenden und dieses Netzwerk nicht überspringen.

C.1.1 Addierer mit EN- und mit ENO-Beschaltung

Hat der Addierer sowohl EN-Beschaltung als auch ENO-Beschaltung, so werden folgende AWL-Befehle abgesetzt:

```
1    U   E   0.0       // EN-Beschaltung
2    SPBNB _001        // VKE ins BIE schieben und springen
                       // wenn VKE == 0
3    L     in1         // Box-Parameter
4    L     in2         // Box-Parameter
5    +I                // Eigentliche Addition
6    T     out         // Box-Parameter
7    UN    OV          // Fehlererkennung
8    SAVE              // Fehler im BIE speichern
9    CLR               // Erstabfrage
10   _001:     U     BIE     // BIE ins VKE schieben
11   =     A   4.0
```

Nach Zeile1 enthält das VKE das Ergebnis der Vorverknüpfung. Der SPBNB-Befehl kopiert das VKE in das BIE und setzt das Erstabfragebit.

- Ist das VKE 0, dann wird in Zeile 10 gesprungen und mit U BIE fortgesetzt. Die Addition wird nicht ausgeführt. In Zeile 10 wird das BIE wieder ins VKE kopiert und dem Ausgang somit 0 zugewiesen.
- Ist das VKE 1, wird nicht gesprungen, d.h. die Addition wird ausgeführt. In Zeile 7 wird ermittelt, ob bei der Addition ein Fehler auftrat, dies wird in Zeile 8 im BIE gespeichert. Zeile 9 setzt das Erstabfragebit. Nun wird in Zeile 10 das BIE-Bit wieder zurück ins VKE kopiert und somit im Ausgang angezeigt, ob die Addition erfolgreich war.
  Das BIE Bit wird durch die Zeilen 10 und 11 nicht mehr verändert, es zeigt also ebenso an, ob die Addition erfolgreich war.

C.1.2 Addierer mit EN- und ohne ENO-Beschaltung

Hat der Addierer eine EN-Beschaltung aber keine ENO-Beschaltung, so werden folgende AWL-Befehle abgesetzt:

```
1    U     E      0.0   // EN-Beschaltung
2    SPBNB _001         // VKE ins BIE schieben und springen
                        //wenn VKE == 0
3    L     in1          // Box-Parameter
4    L     in2          // Box-Parameter
5    +I                 // Eigentliche Addition
6    T     out          // Box-Parameter
7    _001:      NOP   0
```

Nach Zeile1 enthält das VKE das Ergebnis der Vorverknüpfung. Der SPBNB-Befehl kopiert das VKE in das BIE und setzt das Erstabfragebit.

- Ist das VKE 0, dann wird in Zeile 7 gesprungen, die Addition wird nicht ausgeführt, VKE und BIE sind 0.
- War das VKE 1, wird nicht gesprungen, d.h. die Addition wird ausgeführt. Es wird nicht ermittelt, ob bei der Addition ein Fehler auftrat. Das VKE und das BIE sind 1.

C.1.3 Addierer ohne EN- und mit ENO-Beschaltung

Hat der Addierer keine EN-Beschaltung aber ENO-Beschaltung, so werden folgende AWL-Befehle abgesetzt:

```
1    L     in1     // Box-Parameter
2    L     in2     // Box-Parameter
3    +I            // Eigentliche Addition
4    T     out     // Box-Parameter
5    UN    OV      // Fehlererkennung
6    SAVE          // Fehler im BIE speichern
7    CLR           // Erstabfrage
8    U     BIE     // BIE ins VKE schieben
9    =     A      4.0
```

Die Addition wird auf jeden Fall ausgeführt. In Zeile 5 wird ermittelt ob bei der Addition ein Fehler auftrat, dies wird in Zeile 6 im BIE gespeichert. Zeile 7 setzt das Erstabfragebit. Nun wird in Zeile 8 das BIE-Bit wieder zurück ins VKE kopiert und somit im Ausgang angezeigt, ob die Addition erfolgreich war.

Das BIE-Bit wird durch die Zeilen 8 und 9 nicht mehr verändert, es zeigt also ebenso an, ob die Addition erfolgreich war.

C.1.4 Addierer ohne EN- und ohne ENO-Beschaltung

Hat der Addierer weder EN-Beschaltung noch ENO-Beschaltung, so werden folgende AWL-Befehle abgesetzt:

```
1  L     in1     // Box-Parameter
2  L     in2     // Box-Parameter
3  +I            // Eigentliche Addition
4  T     out     // Box-Parameter
5  NOP 0
```

Die Addition wird ausgeführt. Das VKE und das BIE-Bit bleiben unverändert.

C.2 Parameterübergabe

Die Parameter eines Bausteins werden als Wert übergeben. Bei Funktionsbausteinen wird innerhalb des aufgerufenen Bausteins eine Kopie des Aktualparameterwertes im Instanz-DB verwendet. Bei Funktionen liegt eine Kopie des Aktualwertes im Lokaldatenstack. Zeiger werden nicht kopiert. Vor dem Aufruf werden die INPUT-Werte in den Instanz-DB bzw auf den L-Stack kopiert. Nach dem Aufruf werden die OUTPUT-Werte zurück in die Variablen kopiert. Innerhalb des aufgerufenen Baustein arbeitet man nur auf einer Kopie. Die dafür notwendigen AWL-Befehlen befinden sich im aufrufenden Baustein und bleiben dem Anwender verborgen.

Hinweis

Wenn Merker, Eingänge, Ausgänge, Peripherieeingänge oder Peripherieausgänge als Aktualoperanden an einer Funktion verwendet werden, werden diese anders behandelt als die anderen Operanden. Die Aktualisierung erfolgt hier nicht über den L-Stack, sondern direkt.

Ausnahme:

Ist der zugehörige Formalparameter ein Eingangsparameter vom Datentyp BOOL, erfolgt die Aktualisierung der Aktualparameter über den L-Stack.

Warnung

Sorgen Sie bei der Programmierung des aufgerufenen Bausteins dafür, daß die als OUTPUT deklarierten Parameter auch beschrieben werden. Sonst sind die ausgegebenen Werte zufällig! Bei Funktionsbausteinen bekommt man eben den vom letzen Aufruf gemerkten Wert aus dem Instanz-DB, bei Funktionen den zufällig auf dem L-Stack liegenden Wert.

Beachten Sie folgende Punkte:

- Initialisieren Sie wenn möglich alle OUTPUT Parameter.
- Verwenden Sie möglichst keine Setze- und Rücksetze-Befehle. Diese Befehle sind VKE-abhängig. Wenn das VKE den Wert 0 hat, bleibt der zufällige Wert erhalten!
- Wenn Sie innerhalb des Bausteins springen, so achten Sie darauf, daß Sie keine Stellen überspringen, in denen OUTPUT-Parameter beschrieben werden. Denken Sie dabei auch an BEB und die Wirkung der MCR-Befehle.

Index

3

A

B

C

D

E

F

G

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Z