

MELSEC FX-Serie

Speicherprogrammierbare Steuerungen

Programmieranleitung

FX1S / FX1N / FX2N

Zu diesem Handbuch

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele dienen ausschließlich zur Erläuterung der Installation, Bedienung, Programmierung und Anwendung der speicherprogrammierbaren Steuerungen der MELSEC FX1S-, FX1N- und FX2N-Serie.

Die Firma MITSUBISHI ELECTRIC EUROPE B.V. übernimmt auf der Grundlage der Angaben in diesem Handbuch keine Haftung für direkte Schäden oder Folgeschäden, die sich aus dem Gebrauch oder Missbrauch dieser Anleitung ergeben.

Sollten sich Fragen bezüglich Installation und Betrieb der in diesem Handbuch beschriebenen Geräte ergeben, zögern Sie nicht, die Ihnen nächstliegende Niederlassung der MITSUBISHI ELECTRIC EUROPE B.V. zu kontaktieren. Die entsprechenden Adressen finden Sie auf der letzten Seite dieses Handbuchs.

Des weiteren stehen Ihnen folgende Informationsquellen zur Verfügung:

E-Mail: service.ida@meg.mee.com
Hotline: +49 (0)1805 / 000 765 oder 000 766
MEL-FAX-Faxback-System: +49 (0)2102 / 486 485 oder 486 790

Ohne vorherige ausdrückliche schriftliche Genehmigung der Firma MITSUBISHI ELECTRIC EUROPE B.V. dürfen keine Auszüge dieses Handbuchs vervielfältigt, in einem Informationssystem gespeichert oder weiter übertragen werden.

Die Firma MITSUBISHI ELECTRIC EUROPE B.V. behält sich vor, jederzeit technische Änderungen oder Änderungen dieses Handbuchs ohne besondere Hinweise vorzunehmen.

© 03/2001

Programmieranleitung
Speicherprogrammierbare Steuerungen der MELSEC-FX-Familie
FX1S, FX1N und FX2N
Artikel-Nr.: 136748

Version			Änderungen / Ergänzungen / Korrekturen
A	03/2001	pdp	—

Sicherheitshinweise

Zielgruppe

Dieses Handbuch richtet sich ausschließlich an anerkannt ausgebildete Elektrofachkräfte, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut sind. Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Geräte dürfen nur von einer anerkannt ausgebildeten Elektrofachkraft, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut ist, durchgeführt werden.

Bestimmungsgemäßer Gebrauch

Die speicherprogrammierbaren Steuerungen der FX1S-, FX1N- und FX2N-Serie sind nur für die Einsatzbereiche vorgesehen, die in diesem Handbuch beschrieben sind. Achten Sie auf die Einhaltung aller im Handbuch angegebenen Kenndaten.

Es dürfen nur von MITSUBISHI ELECTRIC empfohlene Zusatz- bzw. Erweiterungsgeräte in Verbindung mit den speicherprogrammierbaren Steuerungen der FX1S-, FX1N-, FX2N- und A-Serie benutzt werden.

Jede andere darüber hinausgehende Verwendung oder Benutzung gilt als nicht bestimmungsgemäß.

Sicherheitsrelevante Vorschriften

Bei der Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Geräte müssen die für den spezifischen Einsatzfall gültigen Sicherheits- und Unfallverhütungsvorschriften beachtet werden.

Es müssen besonders folgende Vorschriften (ohne Anspruch auf Vollständigkeit) beachten werden:

- VDE-Vorschriften
 - VDE 0100
Bestimmungen für das Errichten von Starkstromanlagen mit einer Nennspannung bis 1000V
 - VDE 0105
Betrieb von Starkstromanlagen
 - VDE 0113
Elektrische Anlagen mit elektronischen Betriebsmitteln
 - VDE 0160
Ausrüstung von Starkstromanlagen und elektrischen Betriebsmitteln
 - VDE 0550/0551
Bestimmungen für Transformatoren
 - VDE 0700
Sicherheit elektrischer Geräte für den Hausgebrauch und ähnliche Zwecke
 - VDE 0860
Sicherheitsbestimmungen für netzbetriebene elektronische Geräte und deren Zubehör für den Hausgebrauch und ähnliche Zwecke.
- Brandverhütungsvorschriften
- Unfallverhütungsvorschrift
 - VBG Nr.4
Elektrische Anlagen und Betriebsmittel

Erläuterung zu den Gefahrenhinweisen

In diesem Handbuch befinden sich Hinweise, die wichtig für den sachgerechten sicheren Umgang mit dem Gerät sind.

Die einzelnen Hinweise haben folgende Bedeutung:



GEFAHR

Bedeutet, daß eine Gefahr für das Leben und die Gesundheit des Anwenders besteht, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.



ACHTUNG

Bedeutet eine Warnung vor möglichen Beschädigungen des Gerätes oder anderen Sachwerten, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Allgemeine Gefahrenhinweise und Sicherheitsvorkehrungen

Die folgenden Gefahrenhinweise sind als generelle Richtlinie für den Umgang mit der SPS in Verbindung mit anderen Geräten zu verstehen. Diese Hinweise müssen Sie bei der Projektierung, Installation und Betrieb einer Steuerungsanlage unbedingt beachten.



GEFAHR

- Die im spezifischen Einsatzfall geltenden Sicherheits- und Unfallverhütungsvorschriften sind zu beachten. Der Einbau, die Verdrahtung und das Öffnen der Baugruppen, Bauteile und Geräte muß im spannungslosen Zustand erfolgen.
- Baugruppen, Bauteile und Geräte müssen in einem berührungssicheren Gehäuse mit einer bestimmungsgemäßen Abdeckung und Schutzeinrichtung installiert werden.
- Überprüfen Sie spannungsführende Kabel und Leitungen, mit denen die Geräte verbunden sind, regelmäßig auf Isolationsfehler oder Bruchstellen. Bei Feststellung eines Fehlers in der Verkabelung müssen Sie die Geräte und die Verkabelung sofort spannungslos schalten und die defekte Verkabelung ersetzen.
- Überprüfen Sie vor der Inbetriebnahme, ob der zulässige Netzspannungsbereich mit der örtlichen Netzspannung übereinstimmt.
- Treffen Sie die erforderlichen Vorkehrungen, um nach Spannungseinbrüchen und -ausfällen ein unterbrochenes Programm ordnungsgemäß wieder aufnehmen zu können. Dabei dürfen auch kurzzeitig keine gefährlichen Betriebszustände auftreten.
- NOT-AUS-Einrichtungen gemäß VDE 0113 müssen in allen Betriebsarten der Steuerung wirksam bleiben. Ein Entriegeln der NOT-AUS-Einrichtung darf keinen unkontrollierten oder undefinierten Wiederanlauf bewirken.
- Damit ein Leitungs- oder Aderbruch auf der Signalseite nicht zu undefinierten Zuständen in der Steuerung führen kann, sind hard- und softwareseitig entsprechende Sicherheitsvorkehrungen zu treffen.

Inhaltsverzeichnis

1	Einleitung	
1.1	Allgemeines	1-1
1.2	Verwendbare Programmiergeräte/Bediengeräte	1-2
2	Grundlagen der Programmierung	
2.1	Programmverarbeitung in der SPS	2-1
2.1.1	Prozessabbildverfahren	2-2
2.1.2	Signalverarbeitung im Unterschied zur verbindungsprogrammierten Steuerung	2-4
2.2	Steuerungsanweisungen	2-5
2.2.1	Aufbau einer Steuerungsanweisung	2-5
2.2.2	Operanden	2-6
2.2.3	Darstellungsarten von Steuerungsanweisungen	2-7
2.2.4	Zuordnungsliste und Beschaltung der SPS	2-9
3	Operanden	
3.1	Übersicht der Operanden	3-1
3.2	Ein- und Ausgänge	3-2
3.2.1	Ein- und Ausgänge adressieren	3-2
3.2.2	Ein- und Ausgänge programmieren	3-4
3.3	Merker	3-5
3.3.1	Merker adressieren	3-5
3.3.2	Merker programmieren	3-6
3.4	Timer	3-7
3.4.1	Adressierung der Timer	3-7
3.4.2	Programmierung der Timer	3-8
3.4.3	Vorgabe eines Zeitwertes mit den integrierten Potentiometern der Steuerungen der FX1S- und FX1N-Serie	3-9
3.4.4	Genauigkeit der Timer	3-9
3.4.5	Remanente Timer	3-10

3.5	Counter	3-11
3.5.1	16-Bit-Counter	3-12
3.5.2	32-Bit-Counter	3-14
3.5.3	32-Bit-High-Speed-Counter	3-16
3.6	Schrittstatus	3-23
3.6.1	Schrittstatusoperanden adressieren	3-23
3.7	Dezimal-, Hexadezimalkonstanten	3-24
3.7.1	Zahlenwertebereiche der Dezimal-, Hexadezimalkonstanten	3-24
3.8	Register	3-25
3.8.1	Einteilung der Register	3-25
3.8.2	Aufbau der Register	3-26
3.8.3	Adressierung der Register	3-27
3.8.4	Verwendung der Sonderregister	3-27
3.8.5	Extern veränderbare Register	3-28
3.8.6	Einsatz der Indexregister	3-29
3.8.7	Einsatz der File-Register	3-30
3.8.8	Zahlendarstellungen	3-32
3.9	Pointer	3-39
3.9.1	Pointer adressieren	3-39
3.9.2	Nesting-Ebenen	3-39
3.10	Interrupt-Pointer	3-40
3.10.1	Interrupt-Pointer adressieren	3-40
3.11	Nesting	3-45
3.11.1	Nesting-Operanden adressieren	3-45

4 Grundbefehlssatz

4.1	Allgemeine Hinweise	4-1
4.1.1	Erläuterung der Grundbefehlssatztabellen	4-1
4.2	Beginn von Verknüpfungen (LD, LDI)	4-5
4.3	Ausgabe eines Verknüpfungsergebnisses (OUT)	4-7
4.4	UND-Verknüpfungen (AND, ANI)	4-9
4.5	ODER-Verknüpfungen (OR, ORI)	4-11
4.6	Gepulster Beginn von Verknüpfungen (LDP, LDF)	4-13
4.7	Gepulste UND-Verknüpfungen (ANP, ANF)	4-15

4.8	Gepulste ODER-Verknüpfungen (ORP, ORF)	4-17
4.9	UND-Block-Verknüpfung (ANB)	4-19
4.10	ODER-Block-Verknüpfung (ORB).	4-20
4.11	Verarbeiten eines Verknüpfungsergebnisses (MPS, MRD, MPP)	4-21
4.12	Setzen und Rücksetzen einer Kontrollbedingung (MC, MCR)	4-24
4.13	Setzen und Rücksetzen von Operanden (SET, RST)	4-27
4.14	Erzeugen eines einmaligen Impulses (PLS, PLF)	4-29
4.15	Inversion von Verarbeitungsergebnissen (INV)	4-31
4.16	Leerzeile im Programm (NOP)	4-32
4.17	SPS-Programmende (END)	4-34
4.18	Programmbeispiele.	4-35
4.18.1	Abfrage eines Eingangs	4-36

5 STL-Anweisung

5.1	Allgemeine Hinweise	5-1
5.1.1	Anwendungsbeispiel zum Einsatz der STL-Anweisung	5-2
5.1.2	Schematischer Ablauf einer Schrittsteuerung	5-3
5.1.3	Darstellung einer Ablaufsteuerung in einem Flußdiagramm	5-4
5.2	STL-Anweisung programmieren	5-5
5.3	Schrittstatus initialisieren	5-10
5.4	STL-Verzweigungen	5-11
5.4.1	Einfachverlauf.	5-11
5.4.2	Selektive Verzweigung	5-12
5.4.3	Parallele Verzweigung	5-14
5.4.4	Kombination aus selektiver und paralleler Verzweigung	5-16
5.4.5	Leerstatus programmieren	5-17
5.4.6	Sprungverzweigung	5-18
5.5	Beispiel für eine Be- und Entladekontrolle	5-20
5.6	Beispiel für einen Transportier- und Sortiervorgang.	5-22

6	Applikationsanweisungen	
6.1	Allgemeine Hinweise	6-1
6.1.1	Erläuterung zur Beschreibung der Applikationsanweisungen	6-1
6.1.2	Beschreibung der Operanden	6-3
6.1.3	Wortverarbeitung	6-3
6.1.4	Datenstruktur	6-5
6.1.5	Ausführung von Applikationsanweisungen	6-6
6.1.6	Einsatz der Indexregister V, Z	6-7
6.1.7	Bedeutung der Flags	6-8
6.1.8	Programmablauffehler bei der Ausführung von Applikationsanweisungen	6-8
6.1.9	32-Bit-Anweisungen	6-8
6.1.10	Übersicht der Applikationsanweisungen	6-9
6.2	Programmablaufanweisungen	6-11
6.2.1	Sprung innerhalb eines Programms (CJ)	6-12
6.2.2	Aufruf eines Unterprogramms (CALL)	6-16
6.2.3	Ende eines Unterprogramms (SRET)	6-17
6.2.4	Einsatz eines Interrupt-Programms (IRET, EI, DI)	6-18
6.2.5	Ende eines Programmbereichs (FEND)	6-22
6.2.6	Watch-Dog-Timer auffrischen (WDT)	6-23
6.2.7	Programmteile wiederholen (FOR, NEXT)	6-25
6.3	Vergleichs- und Transferanweisungen	6-27
6.3.1	Numerische Daten vergleichen (CMP, DCMP)	6-28
6.3.2	Numerische Datenbereiche vergleichen (ZCP, DZCP)	6-30
6.3.3	Datentransfer (MOV, DMOV)	6-32
6.3.4	Shift-Transfer (SMOV)	6-33
6.3.5	Kopieren und invertieren (CML)	6-36
6.3.6	Block-Transfer (BMOV)	6-37
6.3.7	Transfer von gleichen Daten (FMOV)	6-38
6.3.8	Austausch von Daten (XCH)	6-39
6.3.9	BCD-Konvertierung (BCD, DBCD)	6-41
6.3.10	Binär-Konvertierung (BIN, DBIN)	6-43

6.4	Arithmetische Anweisungen	6-45
6.4.1	Addition numerischer Daten (ADD, DADD)	6-46
6.4.2	Subtraktion numerischer Daten (SUB, DSUB)	6-48
6.4.3	Multiplikation numerischer Daten (MUL, DMUL)	6-50
6.4.4	Division numerischer Daten (DIV, DDIV)	6-52
6.4.5	Inkrementieren (INC, DINC)	6-54
6.4.6	Dekrementieren (DEC)	6-55
6.4.7	Logische UND-Verknüpfung binärer Daten (WAND, DAND)	6-56
6.4.8	Logische ODER-Verknüpfung binärer Daten (WOR, DOR)	6-58
6.4.9	Logische Exklusiv-ODER-Verknüpfung binärer Daten (WXOR, DXOR)	6-60
6.4.10	Negation von Daten (NEG)	6-62
6.5	Verschiebeanweisungen	6-63
6.5.1	Rotation nach rechts (ROR)	6-64
6.5.2	Rotation nach links (ROL)	6-65
6.5.3	Rotieren von Bits nach rechts (RCR)	6-66
6.5.4	Rotieren von Bits nach links (RCL)	6-67
6.5.5	Binäre Daten bitweise verschieben (SFTR, SFTL)	6-68
6.5.6	Daten wortweise nach rechts verschieben (WSFR)	6-70
6.5.7	Daten wortweise nach links verschieben (WSFL)	6-71
6.5.8	Schreiben in einen FIFO-Speicher (SFWR)	6-72
6.5.9	Lesen aus einem FIFO-Speicher (SFRD)	6-73
6.6	Datenoperationen	6-75
6.6.1	Operandenbereiche zurücksetzen (ZRST)	6-76
6.6.2	Daten decodieren (DECO)	6-77
6.6.3	Daten codieren (ENCO)	6-80
6.6.4	Ermittlung gesetzter Bits (SUM)	6-82
6.6.5	Überprüfen eines Bits (BON)	6-83
6.6.6	Ermittlung von Durchschnittswerten (MEAN)	6-84
6.6.7	Starten eines Zeitintervalls (ANS)	6-85
6.6.8	Rücksetzen von Anzeige-Bits (ANR)	6-86
6.6.9	Ermittlung der Quadratwurzel (SQR)	6-87
6.6.10	Umwandlung des Zahlenformats (FLT)	6-88

6.7	High-Speed-Anweisungen	6-89
6.7.1	Ein- und Ausgänge auffrischen (REF)	6-90
6.7.2	Einstellen der Eingangsfiler (REFF)	6-92
6.7.3	Einlesen einer Matrix (MTR)	6-93
6.7.4	Setzen und Rücksetzen durch High-Speed-Counter (DHSCS, DHSCR)	6-96
6.7.5	Bereichsvergleich (DHSZ)	6-98
6.7.6	Geschwindigkeitserkennung (SPD)	6-104
6.7.7	Ausgabe einer definierten Anzahl von Impulsen (PLSY, DPLSY)	6-106
6.7.8	Impulsausgabe mit Modulation der Impulsweite (PWM)	6-108
6.7.9	Ausgabe einer bestimmten Anzahl von Impulsen (PLSR)	6-110
6.8	Anwendungsbezogene Anweisungen	6-112
6.8.1	Schrittstatus initialisieren (IST)	6-113
6.8.2	Suchanweisung (SER)	6-119
6.8.3	Absoluter Counter-Vergleich (ABSD)	6-121
6.8.4	Inkrementaler Counter-Vergleich (INCD)	6-123
6.8.5	Teaching-Timer (TTMR)	6-125
6.8.6	Sonder-Timer (STMR)	6-126
6.8.7	Flip-Flop-Funktion (ALT)	6-127
6.8.8	Rampenfunktion (RAMP)	6-129
6.8.9	Rundtisch-Positionierung (ROTC)	6-131
6.8.10	Sortieranweisung (SORT)	6-134

7 Spezielle FNC-Anweisungen

7.1	Allgemeine Hinweise	7-1
7.1.1	Gesamtübersicht der speziellen FNC-Anweisungen	7-1
7.2	Ein-/Ausgabeeanweisungen	7-4
7.2.1	Zehnertastatur (TKY)	7-5
7.2.2	Hexadezimale Tastatur (HKY)	7-7
7.2.3	Digitaler Schalter (DSW)	7-10
7.2.4	7-Segment-Anzeige (SEGD)	7-12
7.2.5	7-Segment-Anzeige mit Latch (SEGL)	7-13
7.2.6	7-Segment-Anzeige mit zusätzlichen Tasten (ARWS)	7-16
7.2.7	ASCII-Konvertierung (ASC)	7-19
7.2.8	Datenausgabe über die Ausgänge (PR)	7-21
7.2.9	Auslesen von Daten aus einem Sondermodul (FROM)	7-23
7.2.10	Schreiben von Daten in ein Sondermodul (TO)	7-25

7.3	Serielle Kommunikation	7-27
7.3.1	Serielle Datenübertragung (RS)	7-28
7.3.2	Umlegen von Eingängen oder Merkern (PRUN)	7-34
7.3.3	ASCII-Umwandlung (ASCI)	7-35
7.3.4	Hexadezimal-Umwandlung (HEX)	7-37
7.3.5	Summen- und Paritätsprüfung (CCD)	7-39
7.3.6	Einlesen von Sollwerten vom FX1N-8AV-BD und FX2N-8AV-BD (VRRD)	7-41
7.3.7	Einlesen von Schalterstellungen von FX1N-8AV-BD und FX2N-8AV-BD (VRSC)	7-42
7.3.8	Programmierung eines geschlossenen Regelkreises (PID)	7-43
7.4	Anweisung mit Gleitkommazahlen (FX2N)	7-50
7.4.1	Vergleich von Gleitkommazahlen (DECMP)	7-51
7.4.2	Vergleich von Gleitkommazahlen mit einem Bereich (DEZCP)	7-53
7.4.3	Umwandlung des Gleitkommaformats ins wissenschaftliche Zahlenformat (DEBCD)	7-55
7.4.4	Umwandlung wissenschaftliches Zahlenformat ins Gleitkommaformat (DEBIN)	7-56
7.4.5	Addition von Gleitkommazahlen (DEADD)	7-57
7.4.6	Subtraktion von Gleitkommazahlen (DESUB)	7-58
7.4.7	Multiplikation von Gleitkommazahlen (DEMUL)	7-59
7.4.8	Division von Gleitkommazahlen (DEDIV)	7-60
7.4.9	Quadratwurzel aus Gleitkommazahlen (DESQR)	7-61
7.4.10	Umwandlung des Gleitkommaformats ins Dezimal-Format (INT)	7-62
7.4.11	Sinusberechnung mit Gleitkommazahlen (DSIN)	7-63
7.4.12	Cosinusberechnung mit Gleitkommazahlen (DCOS)	7-64
7.4.13	Tangensberechnung mit Gleitkommazahlen (DTAN)	7-65
7.5	Datenverarbeitungsanweisungen	7-66
7.5.1	High-Low-Byte-Tausch (SWAP)	7-67
7.6	Positionieranweisungen (FX1S/FX1N)	7-68
7.6.1	Lesen der absoluten Ist-Position (DABS)	7-69
7.6.2	Referenzpunkt anfahren (ZRN)	7-70
7.6.3	Ausgabe von Impulsen mit variabler Frequenz (PLSV)	7-72
7.6.4	Positionieren auf einen Inkrementalwert (DRVI)	7-74
7.6.5	Positionieren auf einen Absolutwert (DRVA)	7-76
7.7	Echtzeituhr-Anweisungen	7-78
7.7.1	Vergleich von Uhr-Daten (TCMP)	7-79
7.7.2	Vergleich von Uhr-Daten mit einem Bereich (TZCP)	7-81

7.7.3	Addition von Uhr-Daten (TADD)	7-83
7.7.4	Subtraktion von Uhr-Daten (TSUB)	7-85
7.7.5	Lesen von Uhr-Daten (TRD)	7-87
7.7.6	Schreiben von Uhr-Daten (TWR)	7-89
7.7.7	Betriebsstundenzähler (HOUR)	7-90
7.8	Gray-Code-Anweisungen	7-92
7.8.1	Umwandlung Integer in Gray-Code (GRY)	7-93
7.8.2	Umwandlung Gray-Code in Integer (GBIN)	7-94
7.9	Datenaustausch mit Analogmodul FX0N-3A (FX1N)	7-95
7.9.1	Lesen der anlogen Eingangswerte (RD3A)	7-96
7.9.2	Scheiben eines anlogen Ausgangswertes (WR3A)	7-97
7.10	Vergleichsanweisung II	7-98
7.10.1	Lade Vergleiche (LD□)	7-99
7.10.2	UND-verknüpfte Vergleiche (AND□)	7-101
7.10.3	ODER-verknüpfte Vergleiche (OR□)	7-103
8	Sonderfunktionen	
8.1	Datenerhalt im STOP-Modus	8-2
8.2	Betrieb mit konstanter Programmzykluszeit	8-3
8.3	Passwortfunktion	8-4
8.4	Pulse-Catch-Funktion	8-5
8.5	Eingangsfiler einstellen	8-7
8.6	Integrierte Potentiometer bei FX1S und FX1N	8-8
8.7	Echtzeituhr-Funktion	8-9
8.8	File-Register	8-10
8.9	RUN-/STOP-Umschaltung	8-11
8.10	FX2N-Grundgeräte mit 24 V DC-Versorgung	8-11

8.11	Anzeigemodul FX1N-5DM für FX1S und FX1N	8-12
8.11.1	Funktionen.	8-12
8.11.2	Operanden zur Steuerung des Anzeigemoduls.	8-13
8.11.3	Festlegung der angezeigten Operanden.	8-14
8.11.4	Schutz vor unbefugte "Zugriffe".	8-15
8.11.5	Änderung des Zustandes oder der Werte von Operanden	8-15
8.11.6	Automatisches Ausschalten der Beleuchtung der Anzeige	8-16
8.11.7	Freigeben und Sperren der Fehleranzeige	8-16

9 Sondermerker, Sonderregister

9.1	Sondermerker (M8000–M8255)	9-1
9.1.1	SPS-Status (M8000–M8009)	9-2
9.1.2	Zeittakt (M8011–M8019)	9-3
9.1.3	Flags (M8020–M8029)	9-4
9.1.4	SPS-Modus (M8030–M8039)	9-5
9.1.5	STL-Status (M8040–M8049)	9-6
9.1.6	Interrupt-Programm (M8050–M8059)	9-7
9.1.7	Pulse-Catch-Funktion	9-8
9.1.8	Link- und Sonderfunktionen (M8070–M8198)	9-9
9.1.9	Auf-/Abwärts-Counter (M8200–M8254)	9-11
9.2	Sonderregister (D8000–D8195)	9-14
9.2.1	SPS-Status (D8000–D8009)	9-15
9.2.2	Zeittakt (D8010–D8019)	9-16
9.2.3	Flags (D8020–D8029)	9-16
9.2.4	SPS-Modus (D8030 – D8039)	9-17
9.2.5	STL-Status (D8040–D8049)	9-17
9.2.6	Register für Link- und Sonderfunktionen (D8070 – D8099)	9-18
9.2.7	Sonstige Register (D8102 – D8109)	9-18
9.2.8	Register für Kommunikationsadapter (232ADP, 485ADP) (D8120 – D8129)	9-19
9.2.9	Ausführungsregister für HSZ- und PLSY-Anweisungen (D8130 – D8143)	9-20
9.2.10	Index - Register (D8182 – D8195)	9-21

10	Programmfehler	
10.1	Fehlererkennung	10-1
10.1.1	Sondermerker (M8060–M8069)	10-1
10.1.2	Sonderregister (D8060–D8069)	10-2
10.2	Fehlercodes	10-3
10.2.1	Fehlercodes (6101–6409)	10-3
10.2.2	Fehlercodes (6501–6511)	10-4
10.2.3	Fehlercodes (6601–6609)	10-5
10.2.4	Fehlercodes (6610–6632)	10-6
10.2.5	Fehlercodes (6701–6709)	10-7
10.2.6	Fehlercodes (6730–6752)	10-8

A	Technische Daten	
A.1	Übersicht der Grundbefehle	A-1
A.2	Allgemeine Systemdaten MELSEC FX1S	A-4
A.3	Operanden MELSEC FX1S	A-5
A.4	Allgemeine Systemdaten MELSEC FX1N	A-7
A.5	Operanden MELSEC FX1N	A-8
A.6	Applikationsanweisungen MELSEC FX1S/FX1N	A-10
A.7	Allgemeine Systemdaten MELSEC FX2N	A-12
A.8	Operanden MELSEC FX2N	A-13
A.9	Applikationsanweisungen MELSEC FX2N	A-15

B	Ausführungszeiten der Anweisungen	
B.1	Ausführungszeiten FX1S-/FX1N-Serie	B-1
B.1.1	Grundbefehle und Schrittstatus-Anweisungen	B-1
B.1.2	Programmverzweigungsanweisungen	B-3
B.1.3	Vergleichs- und Transferanweisungen	B-3
B.1.4	Arithmetische Anweisungen	B-4
B.1.5	Rotations- und Shift-Anweisungen	B-4
B.1.6	Datenoperationen	B-5
B.1.7	High-Speed-Anweisungen	B-5
B.1.8	Anwendungsbezogene Anweisungen	B-6
B.1.9	Spezielle FNC-Anweisungen	B-7

B.1.10	Positionier-Anweisungen	B-7
B.1.11	Anweisungen für die Echtzeituhr	B-8
B.1.12	Datenaustausch mit Analogeingangsmodule FX0N-3A	B-8
B.1.13	Vergleichsanweisungen	B-8
B.2	Ausführungszeiten FX2N	B-11
B.2.1	Grundbefehle und Schrittstatus-Anweisungen	B-11
B.2.2	Programmverzweigungsanweisungen	B-13
B.2.3	Vergleichs- und Transferanweisungen	B-13
B.2.4	Arithmetische Anweisungen	B-14
B.2.5	Rotations- und Shift-Anweisungen	B-15
B.2.6	Datenoperationen	B-16
B.2.7	High-Speed-Anweisungen	B-17
B.2.8	Anwendungsbezogene Anweisungen	B-17
B.2.9	Spezielle FNC-Anweisungen	B-18
	Index	B-21

1 Einleitung

1.1 Allgemeines

Anwendungsbereich

Das vorliegende Handbuch beschreibt und erläutert sämtliche Operanden sowie die Adressenzuordnung von Ein- und Ausgängen, die zur Programmierung der speicherprogrammierbaren Steuerungen der MELSEC FX1S-, FX1N- und FX2N-Serie notwendig sind.

Das Handbuch enthält die elementaren Grundlagen zur Programmierung der MELSEC FX1S, FX1N und FX2N. Als weiterführende Ergänzung hierzu dienen die jeweiligen Handbücher zu den unterschiedlichen Sondermodulen.

Die beschriebenen Operanden, Adressen, Anweisungen und Programme beziehen sich auf folgende Steuerungstypen:

- MELSEC FX1S
- MELSEC FX1N
- MELSEC FX2N

HINWEISE

Wenn nicht besonders gekennzeichnet, betreffen alle allgemeingültigen Aussagen immer alle genannten Steuerungstypen. Die im Verlauf des Handbuches verwendete Bezeichnung „FX-Familie“ bezieht sich daher grundsätzlich auf alle 3 Steuerungstypen.

Angaben oder Besonderheiten, die immer jeweils nur einen bestimmten Steuerungstyp betreffen, sind entsprechend gekennzeichnet.

Informationen zur Installation, Inbetriebnahme, Wartung und Fehlerbehebung der Geräte sind den entsprechenden Hardware-Handbüchern zu entnehmen.

1.2 Verwendbare Programmiergeräte/Bediengeräte

In der folgenden Tabelle sind die Programmiergeräte aufgeführt, die direkt mit den Anweisungen der FX2-Familie arbeiten können.

Beschreibung	Modell	Neue Version, völlig kompatibel
Handprogrammiergerät	FX-10P-E	FX2N ab V 3.00 FX1N/FX1S ab V 4.00
Handprogrammiergerät	FX-20P-MFXA-E	FX2N ab V 3.00 FX1N/FX1S ab V 5.00
Datenverarbeitungseinheit	FX-10DU-E	V 4.00
	FX-20DU-E	Unterstützt nur die Operanden der FX- Serie
	FX-30DU-E	V 3.00
	FX-40DU-E(S)	Unterstützt nur die Operanden der FX- Serie
	FX-40DU-TK-ES	V 3.00
	FX-50DU-TK(S)-E	V 2.10

Außerdem können die Bediengeräte der F-GOT-, A-GOT- und MAC E-Serie verwendet werden.

2 Grundlagen der Programmierung

2.1 Programmverarbeitung in der SPS

Funktionsprinzip

Über die Eingänge einer SPS werden analoge oder binäre Signale aufgenommen, über das SPS-Programm verarbeitet und an die nachgeordneten Ausgänge der SPS abgegeben.

Arbeitsweise

Eine SPS arbeitet nach einem vorgegebenen Programm. Ein solches Programm wird in die SPS übertragen und im Programmspeicher abgelegt.

Das Programm besteht aus einer Folge einzelner Steuerungsanweisungen, die die Funktion der Steuerung festlegen. Die SPS arbeitet die Steuerungsanweisungen entsprechend der programmierten Reihenfolge nacheinander - also seriell - ab. Zur Erstellung eines Steuerungsprogramms muß der eigentliche Steuerungsprozeß daher in einzelne Anweisungen zerlegt werden.

Der gesamte Programmdurchlauf wird ständig wiederholt, es findet also ein zyklischer Programmdurchlauf statt. Die für einen Programmdurchlauf benötigte Zeit wird als Programmzykluszeit bezeichnet.

2.1.1 Prozessabbildverfahren

Das Programm wird in der SPS nach dem sogenannten Prozessabbildverfahren abgearbeitet. Die folgende Abb. zeigt die Verarbeitung eines Programms nach diesem Verfahren.

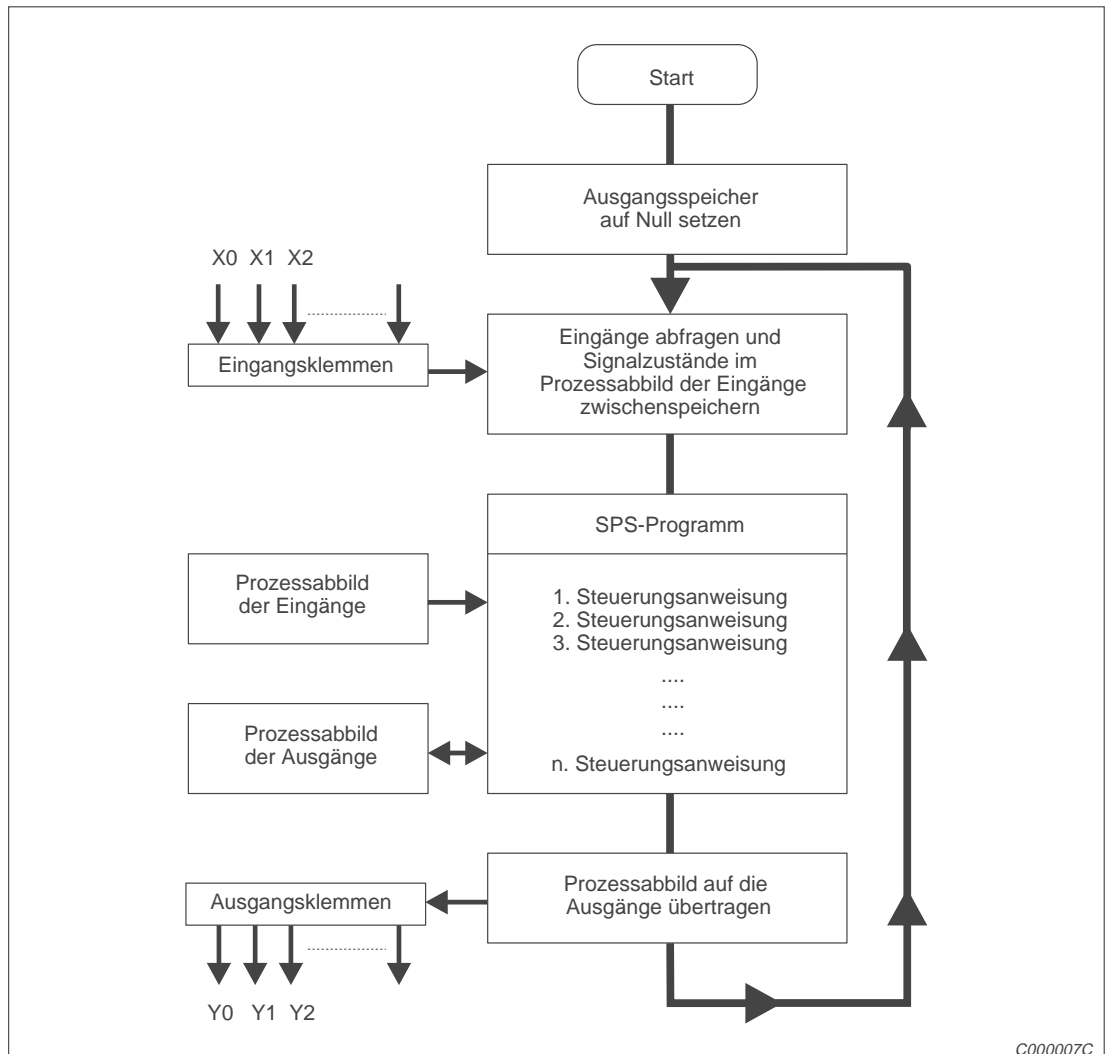


Abb. 2-1: Abarbeitung eines Programms nach dem Prozessabbildverfahren

Prozessabbild der Eingänge

Nach dem Programmstart schaltet die SPS die Ausgänge in einen definierten Ausgangszustand. Anschließend werden die Signalzustände der Eingänge abgefragt und in einem Speicher hinterlegt. Es wird ein Prozessabbild der Eingänge erstellt.

Programmdurchlauf

Während des anschließenden Programmdurchlaufs greift der Mikroprozessor auf die gespeicherten Eingangszustände im Prozessabbild zu. Jede Steuerungsanweisung wird dann entsprechend ihrer Reihenfolge mit den Operanden verknüpft. Das Verknüpfungsergebnis wird zwischengespeichert. D. h., Signaländerungen an den Eingängen werden erst im nächsten Programmzyklus erkannt.

Prozessabbild der Ausgänge

Verknüpfungen, die die Ausgänge betreffen, werden jeweils in einem Ausgangszwischenspeicher hinterlegt (Prozessabbild der Ausgänge).

Erst am Ende des Programmdurchlaufs werden die Zwischenergebnisse an die Ausgänge übertragen. Im Ausgangszwischenspeicher liegt immer ein Prozessabbild der Ausgänge vor. Beim Programmdurchlauf wird nie auf die Ein- und Ausgänge direkt, sondern nur auf ihr Prozessabbild zugegriffen. Nach der Wertzuweisung an die Ausgänge wird der Programmzyklus wiederholt.

2.1.2 Signalverarbeitung im Unterschied zur verbindungsprogrammierten Steuerung

Bei einer verbindungsprogrammierten Steuerung werden alle Steuerungsvorgänge gleichzeitig (parallel) ausgeführt. Jede Änderung der Eingangssignalzustände bewirkt sofort eine Änderung der Ausgangssignalzustände.

HINWEIS

Bei einer SPS kann eine Änderung der Eingangssignalzustände während des Programmdurchlaufs erst wieder beim nächsten Programmzyklus berücksichtigt werden. Dieser Nachteil wird durch entsprechend kurze Zykluszeiten weitgehend wieder ausgeglichen.

Die Zykluszeit ist abhängig von der Anzahl der Steuerungsanweisungen und von der Art der eingesetzten Steuerungsanweisungen.

2.2 Steuerungsanweisungen

Das SPS-Programm besteht aus einer Folge von Verknüpfungen, die die Funktion der Steuerung festlegen. Zur Erstellung des Programms ist es daher notwendig, die Steuerungsaufgabe in einzelne Steuerungsanweisungen zu zerlegen. Eine Steuerungsanweisung ist die kleinste Einheit eines Programms.

2.2.1 Aufbau einer Steuerungsanweisung

Eine Steuerungsanweisung besteht aus einer Schrittnummer, einer Anweisung (Befehl) und einem Operanden.

Steuerungsanweisung			
Schritt- nummer	Anweisung (Befehl)	Operand	
		Operanden- kennzeichen	Operanden- adresse
„015“	„AND“	„Y“	„003“

Tab. 2-1:
Aufbau einer Steuerungsanweisung

- Die Anweisungen werden in einer bestimmten Reihenfolge abgearbeitet, die durch die Angabe der Schrittnummer festgelegt wird.
- Die Anweisung (Befehl) beschreibt die auszuführende Funktion, also die Art der Verknüpfung.
- Der Operand gibt an, womit eine Verknüpfung (Anweisung) ausgeführt werden soll. Ein Operand kann zum Beispiel eine Eingangsklemme, eine Ausgangsklemme oder ein interner Zähler sein.

HINWEIS

Bei bestimmten Steuerungsanweisungen (Befehlen) kann die Angabe des Operanden und/oder der Operandenadresse entfallen.

2.2.2 Operanden

Der Operand besteht aus einem

- Operandenkennzeichen und
- einer Operandenadresse.

Das Operandenkennzeichen definiert die Art des Operanden.

Die Angabe der Operandenadresse ermöglicht

- eine Unterscheidung bei einer mehrfachen Benutzung des gleichen Operandenkennzeichens oder
- die Festlegung von Zahlenwerten z. B: für Konstanten.

Die folgende Tabelle enthält eine Übersicht aller programmierbaren Operanden und deren entsprechenden Operandenkennzeichen.

Operand	Operandenkennzeichen	Bedeutung
Eingang	X	Eingangsklemme der SPS
Ausgang	Y	Ausgangsklemme der SPS
Merker	M	Hilfsrelais Speicher für binäre Zwischenergebnisse
Timer	T	Zeitglied Speicher zur Realisierung von Zeiten
Counter	C	Zähler Speicher zur Realisierung von Zählern
Schrittstatus	S	Festgelegter Schritt Programmierung von Ablaufsteuerungen
Dezimalkonstante	K	Festgelegter dezimaler Zahlenwert
Hexadezimalkonstante	H	Festgelegter hexadezimaler Zahlenwert
Datenregister	D	Datenspeicher 16-Bit-, 32-Bit-Format
Indexregister	V, Z	Datenspeicher für Zwischenergebnisse, Indizierung 16-Bit-Format
Pointer	P	Sprungzieladresse Markierung für einen Programmsprung
Interrupt-Pointer	I	Programmunterbrechung Sprung zum Interrupt-Programm
Nesting	N	Programmverzweigung

Tab. 2-2: Operanden und die zugehörigen Operandenkennzeichen

2.2.3 Darstellungsarten von Steuerungsanweisungen

Eine SPS-Programmierung kann in drei verschiedenen Darstellungsarten erfolgen:

- Anweisungsliste (AWL)
- Funktionsplan (FUP)
- Kontaktplan (KOP)

HINWEISE

Abhängig vom genutzten Programmiersystem können nicht alle drei Darstellungsarten genutzt werden.

Eine Programmierung mit den in den IEC 1131.3 definierten Darstellungsarten ist ebenfalls möglich.

Anweisungsliste

Die Anweisungsliste stellt das Programm als eine Abfolge von Steuerungsanweisungen in einer Liste dar.

Schritt- nummer	Anweisung	Operanden- kennzeichen	Operanden- adresse
000	LD	X	000
001	ORI	X	001
002	OUT	Y	000
003	END	—	—

Tab. 2-3: Beispiel einer Anweisungsliste

Funktionsplan

Der Funktionsplan stellt das Programm als eine Abfolge von Netzwerken dar, wobei die Steueranweisungen innerhalb der Netzwerke als Funktionsblöcke erscheinen.

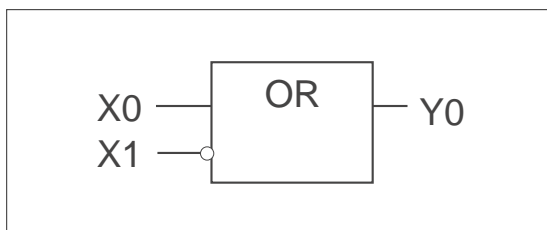




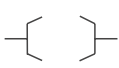
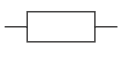
Abb. 2-2: Beispiel eines Funktionsplans

C000381C

Kontaktplan

Der Kontaktplan ist eine Anlehnung an den Stromlaufplan in aufgelöster Darstellung. Entgegen der dort üblichen senkrechten Anordnung der Strompfade werden im Kontaktplan die Strompfade waagrecht dargestellt und untereinander angeordnet.

Im wesentlichen werden die folgenden vier Grundsymbole verwendet.

Symbol	Bedeutung
	Symbol für einen Signaleingang mit Abfrage auf Signalzustand „1“
	Symbol für einen Signaleingang mit Abfrage auf Signalzustand „0“
	Symbol für einen Signalausgang. Bei Ansteuern mit einem „1“-Signal wird ein „1“-Signal an den entsprechenden Operanden zugewiesen
	Symbol für Sonderfunktionen

Tab. 2-4:
Kontaktplansymbolik

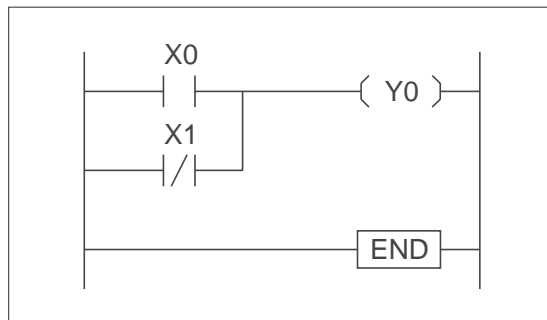


Abb. 2-3:
Beispiel eines Kontaktplans

C000004C

2.2.4 Zuordnungsliste und Beschaltung der SPS

Neben der Beschreibung des SPS-Programms ist für die Praxis eine Zuordnungsliste und die Beschaltung der SPS wichtig.

Zuordnungsliste

Aus der Zuordnungsliste wird ersichtlich

- mit welchen Geräten die Ein- und Ausgänge beschaltet sind,
- welche in der SPS vorhandenen Funktionen (Zähler, Merker, usw.) für den Steuerungsprozess eingesetzt werden.

Benennung	Stromlaufplan- kennzeichen	Operanden- kennzeichen	Operanden- adresse
Schließer „Ein“	S1	X	000
Öffner „Aus“	S2	X	001
Melder	H1	Y	000
Melder	H2	Y	001
Timer (100 ms)	—	T	003

Tab. 2-5: Beispiel einer Zuordnungsliste

Beschaltung der SPS

Die Beschaltung der SPS stellt die Verbindungen zwischen der SPS und den angeschlossenen Ein- und Ausgabegeräten dar.

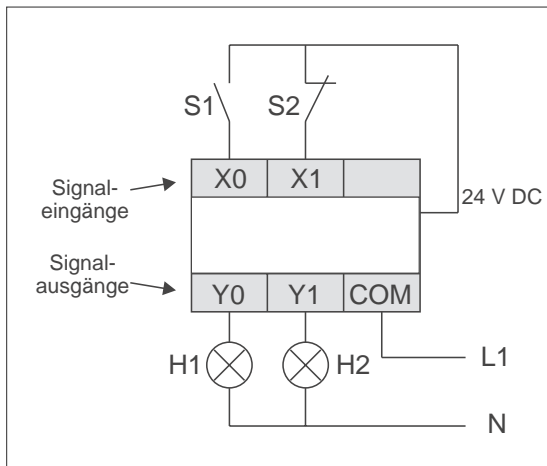


Abb. 2-4: Beispiel einer SPS-Beschaltung

C000005C

3 Operanden

3.1 Übersicht der Operanden

Dieses Kapitel beschreibt alle verfügbaren Operanden und deren Einsatzmöglichkeiten innerhalb eines SPS-Programms. Mit der Angabe eines Operanden wird festgelegt, womit eine Operation (Anweisung) durchgeführt wird.

Ein Operand besteht aus einem

- Operandenkennzeichen und
- einer Operandenadresse.

MELSEC-Operanden		Max. Anzahl der Operanden		
		FX1S	FX1N	FX2N
Eingang	X	maximal 16, abhängig von der Art der Steuerung	Summe bis 128	Summe bis 256
Ausgang	Y	maximal 14, abhängig von der Art der Steuerung		
Merker	M	384	384	3072
Timer	T	64	256	256
Counter	C	53	256	256
High-Speed-Counter (gleichzeitig nutzbar)	C	21	21	21
Schrittstatus	S	128	1000	1000
dez. Konst.	K	16/32 Bit	16/32 Bit	16/32Bit
hex. Konst.	H	16/32 Bit	16/32 Bit	16/32Bit
Datenregister	D	256	8000	8000
File-Register	D	—	7000 (anteilig)	7000 (anteilig)
Indexregister	V, Z	16	16	16
Pointer	P	64	128	128
Interrupt-Pointer	I	6	6	6 Eingänge 3 Timer
Nesting	N	8	8	8

Tab. 3-1: Operanden und Operandenkennzeichen

Eine detaillierte Übersicht der Operanden und Operandenadressen für jeden Steuerungstyp befindet sich im Anhang dieses Handbuchs.

3.2 Ein- und Ausgänge

Die Ein- und Ausgänge werden im SPS-Programm durch Operanden dargestellt. Durch die Angabe einer zusätzlichen Operandenadresse können Sie gezielt die einzelnen Ein- und Ausgänge beim Programmieren ansprechen.

3.2.1 Ein- und Ausgänge adressieren

Die Adressierung der Ein- und Ausgänge muß oktal erfolgen, d. h. es findet ein Stellensprung schon nach 8 Ziffern statt (0,1,2,3,4,5,6,7,10,11,...,16,17).

Gerätetyp	Art der Operanden	Operandenkennzeichen, Operandenadressen	Anzahl der Adressen
FX1S	Eingänge	X0 bis X17	6 – 16
	Ausgänge	Y0 bis Y15	4 – 14
FX1N	Eingänge	X0 bis X157	max. 128 ^①
	Ausgänge	Y0 bis Y157	max. 128 ^①
FX2N	Eingänge	X0 bis X317	max. 256 ^②
	Ausgänge	Y0 bis Y317	max. 256 ^②

Tab. 3-2: Anzahl der vorhandenen Ein- und Ausgänge mit den zugehörigen Operandenadressen

- ① Die Summe der Ein-/Ausgänge (Hardware) ist 128. Durch die Software können 128 Eingänge und 128 Ausgänge adressiert werden.
- ② Die Summe der Ein-/Ausgänge (Hardware) ist 256. Durch die Software können 256 Eingänge und 256 Ausgänge adressiert werden.

Verarbeitung von Eingangssignalen mit kurzen Impulszeiten

Sehr kurze Eingangsimpulse werden nicht berücksichtigt. Die Ein- und Ausschaltsignale von Eingängen müssen während des gesamten Programmzyklus anstehen.

Bei einer Programmzykluszeit von z. B. 10 ms und einer Schaltverzögerung von 10 ms müssen die Ein- und Ausschaltzeiten von Eingängen länger als 20 ms sein.

Bei diesem Beispiel können Eingangssignale, deren Schaltfrequenz mehr als 25 Hz beträgt, nicht direkt verarbeitet werden. Eine Programmverarbeitung dieser Signale ist mit Hilfe von Applikationsanweisungen möglich.

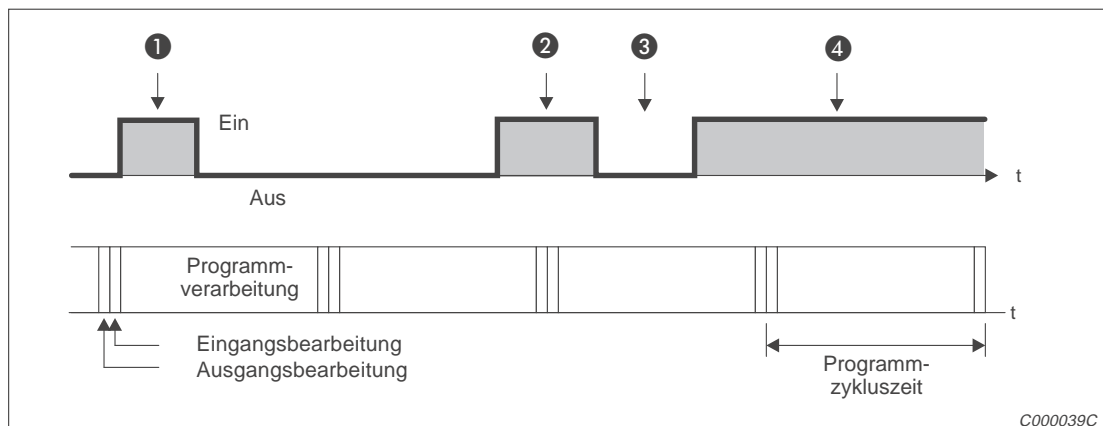


Abb. 3-1: Verarbeitung von Eingangssignalen mit kurzen Impulszeiten

- ① und ③: dieser Eingangsstatus wird nicht erkannt;
- ②: dieser Eingangsstatus wird zufällig erkannt;
- ④: dieser Eingangsstatus wird immer korrekt erkannt.

3.2.2 Ein- und Ausgänge programmieren

Die Signalzustände der Ein- und Ausgänge können mit verschiedenen Anweisungen im Programm abgefragt werden.

Über die Ausgänge können Verknüpfungsergebnisse ausgegeben werden. Zusätzlich lassen sich die Signalzustände der Ausgänge im Programm direkt festlegen (Setzen oder Zurücksetzen).

Beispiel ▾

Einsatz der Ein- und Ausgänge

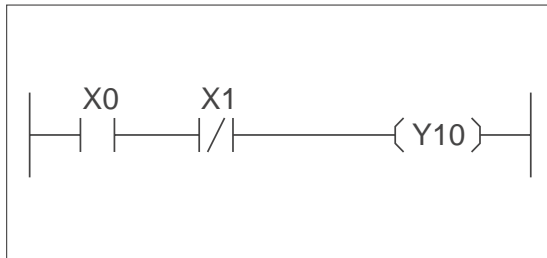


Abb. 3-2:

Programmierbeispiel zum Einsatz von Ein- und Ausgängen

C000122C

Der Ausgang Y10 weist den Signalzustand „1“ auf, wenn folgende Bedingungen erfüllt sind:

- Der Eingang X0 weist den Signalzustand „1“ auf,
und
- der Eingang X1 weist den Signalzustand „0“ auf.

△

HINWEIS

Das Relais oder der Transistor des Ausgangs Y10 wird nach der Abarbeitung des SPS-Zyklus eingeschaltet.

3.3 Merker

Zum Speichern von binären Verknüpfungsergebnissen (Signalzustand „0“ oder „1“) innerhalb eines Programms werden Zwischenspeicher (Merker) eingesetzt. Diese Merker entsprechen in der Verwendung den Hilfsrelais in den Relaissteuerungen.

Die FX-Familie stellt neben den „normalen Merkern“ auch sogenannte Latch- und Sondermerker zur Verfügung.

- Latch-Merker behalten auch bei einem Spannungsausfall ihre Informationen. Die Informationen werden in einem spannungsausfallsicheren Speicher zwischengespeichert.
- Sondermerker stellen spezielle Sonderfunktionen zur Verfügung (siehe Abs. 10.1).

3.3.1 Merker adressieren

Die Adressierung der Merker und Latch-Merker muß dezimal erfolgen.

Steuerung	Operand	Speicherinhalt im EEPROM gesichert	Operandenkennzeichen Operandenadressen	Anzahl der Adressen
FX1S	Merker	—	M0 – M383	384
	Latch-Merker	●	M384 – M511	128
	Sondermerker	●	M8000 – M8255	256
FX1N	Merker	—	M0 – M383	384
	Latch-Merker	●	M384 – M1535	1152
	Sondermerker	●	M8000 – M8255	256
FX2N	Merker ^①	—	M0 – M3071	3072
	Latch-Merker	—	M500 – M3071	2572 (anteilig)
	Sondermerker	—	M8000 – M8255	256

Tab. 3-3: Merker und die zugehörigen Operandenadressen

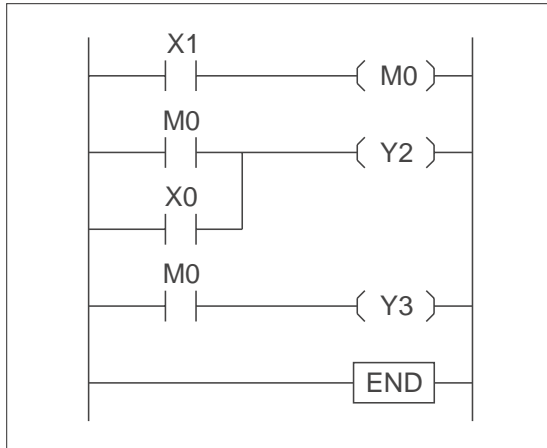
- ① Die Merker M2800 bis M3071 können in Verbindung mit gepulsten Anweisungen (LDP, LDF etc.) als flankengesteuerte Merker verwendet werden.

3.3.2 Merker programmieren

Merker werden wie Ausgänge programmiert. Es besteht aber keine Möglichkeit, an diese Merker außerhalb der SPS Geräte anzuschließen, weil Merker nur Speicherstellen im Arbeitsspeicher der SPS darstellen.

Beispiel ▾

Einsatz der Merker

**Abb. 3-3:***Beispiel zum Einsatz von Merkern*

C000024C

Wenn der Eingang X1 den Signalzustand „1“ aufweist, schaltet der Merker M0 auf den Signalzustand „1“. Der Merker M0 schaltet dann die Ausgänge Y2 und Y3 auf den Signalzustand „1“. Wenn der Eingang X0 den Signalzustand „1“ aufweist, wird der Ausgang Y2 unabhängig von M0 auf den Signalzustand „1“ geschaltet. △

3.4 Timer

Für einige Steuerungsprozesse, wie z. B. einem zeitabhängigen Zuschalten eines Lüftungsmotors, werden Zeitglieder benötigt. In der Relais-technik werden hierfür Zeitrelais mit Einschalt- oder Ausschaltverzögerung eingesetzt. Die SPS-Technik verwendet interne Zeitglieder, deren Verhalten durch das Programm bestimmt werden kann.

Es wird unterschieden zwischen analogen und digitalen Zeitgliedern (Timern). Zum Starten und Rücksetzen werden Timer wie Ausgänge programmiert. Der Schaltkontakt eines Timers kann innerhalb des Programmes beliebig oft abgefragt werden.

3.4.1 Adressierung der Timer

Die Timer werden dezimal adressiert.

Steuerung	Operandenkennzeichen	Anzahl der Adressen	Zeitschritte	Zeitbereich	Sondermerker M8028
FX1S	T0 – T62	63	100 ms	0 – 3276,7 s	Aus
	T0 – T31	32	100 ms	0 – 3276,7 s	Ein
	T32 – T62	31	10 ms	0 – 327,67 s	Ein
	T63	1	1 ms	0 – 32,767 s	—
FX1N	T0 – T199	200	100 ms	0 – 3276,7 s	—
	T200 – T245	46	10 ms	0 – 327,67 s	—
	T246 – T249	4	1 ms	0 – 32,767 s	—
	T250 – T255 (remanent)	6	100 ms	0 – 3276,7 s	—
FX	T0 – T199	200	100 ms	0 – 3276,7 s	—
	T200 – T245	46	10 ms	0 – 327,67 s	—
	T246 – T249	4	1 ms	0 – 32,767 s	—
	T250 – 255 (remanent)	6	100 ms	0 – 3276,7 s	—

Tab. 3-4: Einstellbarer Zeitbereich der Timer und deren zugehörige Operandenadressen

HINWEIS

Wenn bei einer Steuerung der FX1S-Serie mit den Versionsnummern V1.00 bis V1.30 der Timer T63 (Zeitschritt 1 ms) im Programm verwendet wird, muss das Programm um die folgende Anweisung erweitert werden. Die Anweisung braucht nur einmal, vor dem Zugriff auf T63, eingefügt werden. Bei Steuerungen ab Version 1.40 ist diese Anweisung nicht notwendig.

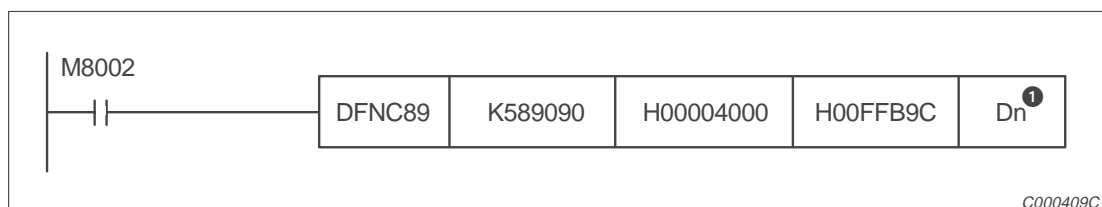


Abb. 3-4: Programmiererweiterung bei Verwendung von T63 bei einer FX1S der Versionen V1.00 bis V1.30

- ① Die Nummer des Registers wird vom Anwender festgelegt. Durch die Anweisung werden zwei aufeinanderfolgende Register belegt.

3.4.2 Programmierung der Timer

Der gewählte Zeitsollwert wird durch eine zusätzliche Dezimalkonstante K festgelegt, die die Anzahl der Zeitschritte angibt.

Beispiel ▾ Bei einem 100-ms-Timer, bei dem die Dezimalkonstante mit $K = 5$ definiert ist, entspricht dies einem Zeitwert von $5 \times 100 \text{ ms} = 500 \text{ ms}$.



Die Timer arbeiten als Einschaltverzögerung. Ein Timer wird durch Ansteuern mit einem „1“-Signal aktiviert. Nach Ablauf des eingestellten Zeitsollwertes schaltet der Timer auf den Signalzustand „1“. Ein Timer fällt in den Ruhezustand zurück, sobald kein „1“-Signal an seinem Eingang mehr ansteht.

HINWEIS | Die Angabe des Zeitsollwertes kann auch indirekt über den in einem Datenregister gespeicherten dezimalen Zahlenwert vorgenommen werden.

Beispiel ▾ Das folgende Kontaktplanbeispiel zeigt den Einsatz der Timer und die indirekte Festlegung des Zeitsollwertes.

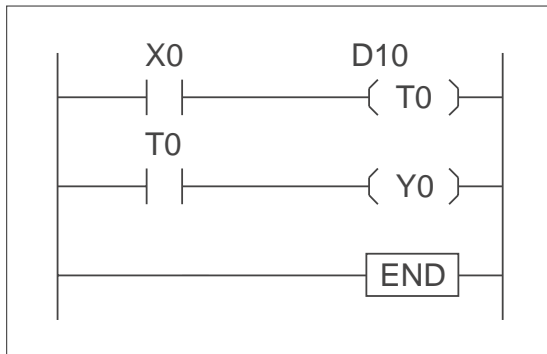


Abb. 3-5:
Programmierbeispiel zum Einsatz der Timer

C000201C

Der Zeitsollwert ist im Datenregister D10 abgespeichert.



Beispiel ▾ Das folgende Kontaktplanbeispiel zeigt den Einsatz der Timer und die direkte Festlegung des Zeitsollwertes.

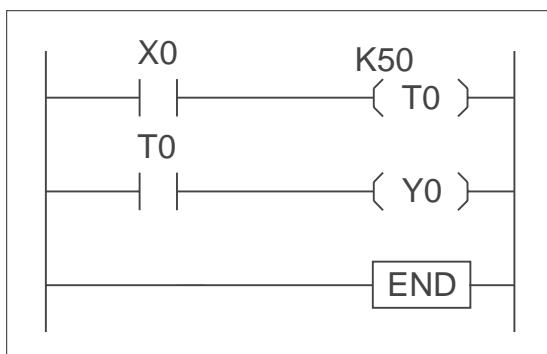


Abb. 3-6:
Vorgabe des Zeitsollwertes durch eine Konstante

C000027C

Wenn der Eingang X0 den Signalzustand „1“ aufweist, beginnt die eingestellte Zeit abzulaufen. Nach Ablauf der programmierten Zeit $t = 5 \text{ s}$ wird der Ausgang Y0 auf den Signalzustand „1“ geschaltet. Der Timer T0 fällt in den Ruhezustand zurück, sobald der Eingang X0 den Signalzustand „0“ aufweist.



3.4.3 Vorgabe eines Zeitwertes mit den integrierten Potentiometern der Steuerungen der FX1S- und FX1N-Serie

Über zwei Drehpotentiometer kann der Datenwert in zwei Sonderdatenregistern manuell im Bereich von 0 bis 255 variiert werden. Der Wert des oberen Potentiometers VR1 kann aus dem Sonderdatenregister D8030 gelesen werden, das untere Potentiometer VR2 beeinflusst den Inhalt des Datenregisters D8031. Der Inhalt des Datenregisters kann dann als Sollwertvorgabe für Timer, aber auch für Counter (Zähler) im Programm verwendet werden.

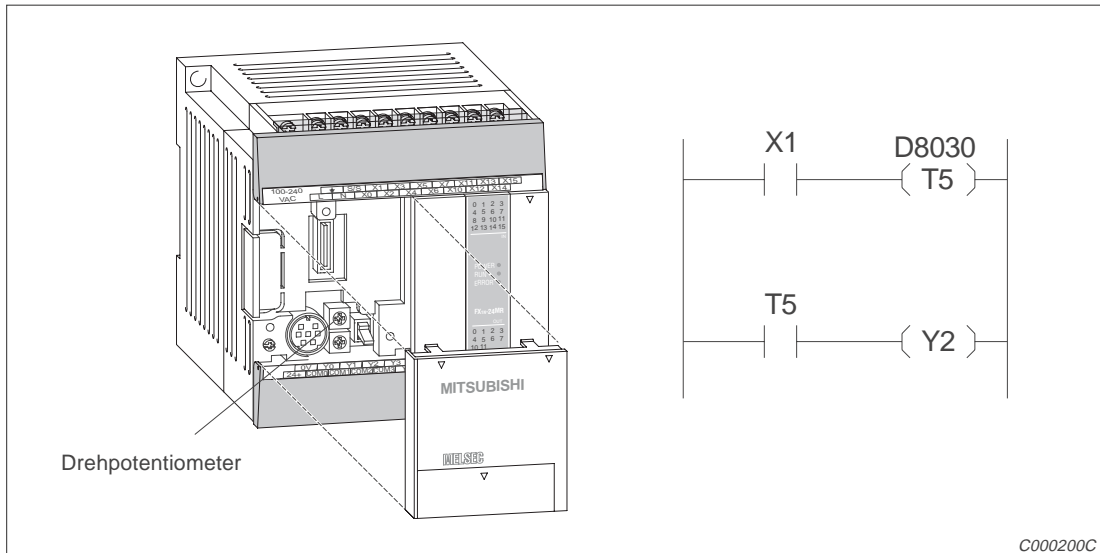


Abb. 3-7: Beispiel zur Einstellung eines Zeitsollwertes im Datenregister D8030 mit einem Potentiometer

Bei einer Steuerung der FX2N-Serie ist diese Art der Sollwertvorgabe optional möglich.

Steuerung	Anzahl der Potentiometer	Zugehöriges Datenregister
FX1S FX1N	2	D8030 (VR1) D8031 (VR2)
FX2N	optional 8	siehe Applikationsanweisung VRSC

Tab. 3-5:
Zuordnung der Potentiometer

3.4.4 Genauigkeit der Timer

Der Ablaufvorgang eines Timers beginnt, sobald die Eingangsbedingung gesetzt ist.

Die Genauigkeit der Timer beträgt: $(T - \alpha) \leq T \leq (T + T_0)$

T: Zeitsollwert

T0: Programmzykluszeit

α : Timer-Zeitschritt (100 ms, 10 ms, 1 ms)

Befindet sich die Setzanweisung des Timerarbeitskontaktes im Programm vor der Festlegung des Timers, kann die Verzögerung maximal (+2 T0) betragen.

Beträgt der Zeitsollwert T = 0, wird der Timerarbeitskontakt eingeschaltet, sobald im nächsten Programmzyklus die entsprechende Setzanweisung abgearbeitet wird.

3.4.5 Remanente Timer

Die Steuerungen der FX1N- und FX2N-Serie verfügen neben den bereits beschriebenen Zeitgliedern auch über remanente Zeitglieder (T246 bis T255), die auch nach dem Abschalten der ansteuernden Verknüpfung den bereits erreichten Zeitistwert behalten.

Die Zeitistwerte werden in einem Speicher abgelegt, dessen Inhalt auch bei einem Spannungsausfall erhalten bleibt.

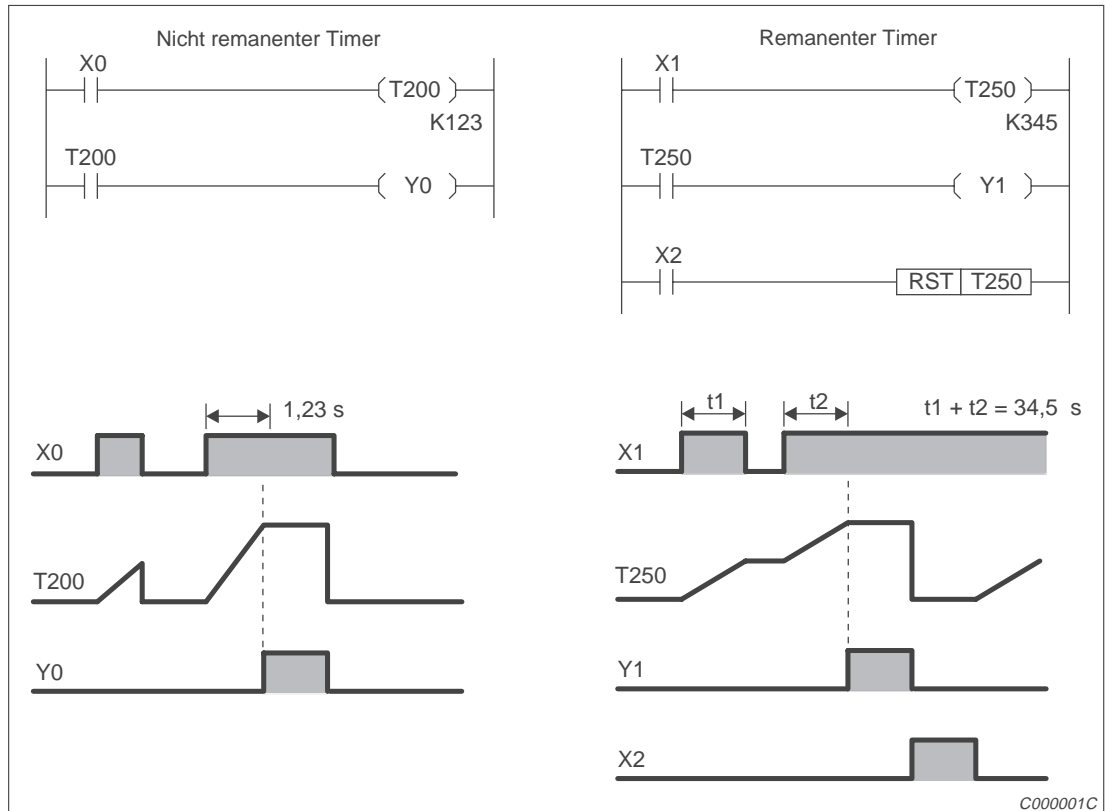


Abb. 3-8: Arbeitsweise der remanenten Timer

3.5 Counter

Damit Sie Zählvorgänge programmieren können, stellt Ihnen die FX-Familie mehrere interne Zähler (Counter) zur Verfügung.

Die Counter lassen sich in folgende Gruppen einteilen:

- 16-Bit-Counter, aufwärtszählend
Es werden programminterne Zählsignale verarbeitet. Der Zähleristwert bleibt bei einem Spannungsausfall der SPS nicht gespeichert.
- 16-Bit-Counter, aufwärtszählend
Es werden programminterne Zählsignale verarbeitet. Der Zähleristwert wird in einem spannungsausfallsicheren Speicher abgespeichert und bleibt bei einem Spannungsausfall der SPS erhalten.
- 32-Bit-Counter, auf-/ abwärtszählend
Es werden programminterne Zählsignale verarbeitet. Die Zählrichtung kann durch den Zustand eines Sondermerkers beeinflusst werden. Die Zähleristwerte bleiben bei einem Spannungsausfall der SPS nicht gespeichert.
- 32-Bit-Counter, auf-/ abwärtszählend
Es werden programminterne Zählsignale verarbeitet. Die Zählrichtung kann durch den Zustand eines Sondermerkers beeinflusst werden. Der Zähleristwert wird bei einem Spannungsausfall der SPS in einem spannungsausfallsicheren Speicher abgelegt.
- 32-Bit-High-Speed-Counter (schnelle Zähler), auf-/ abwärtszählend
High-Speed-Counter verarbeiten sehr schnell aufeinanderfolgende **externe** Zählsignale unabhängig von der benötigten Programmzykluszeit.

3.5.1 16-Bit-Counter

16-Bit-Counter adressieren

Die Adressierung der 16-Bit-Counter erfolgt dezimal.

Steuerung	Operandenadresse	Anzahl	Spannungsausfallsicher
FX1S	C0 – C15	16	—
	C16 – C31	16	●
FX1N	C0 – C15	16	—
	C16 – C199	184	●
FX2N	C0 – C99	100	—
	C100 – C199	100	●

Tab. 3-6: 16-Bit-Counter und zugehörige Operandenadressen

16-Bit-Counter programmieren

Der gewählte Zählersollwert wird durch eine zusätzliche Dezimalkonstante K festgelegt. Für die Dezimalkonstante K kann ein Zahlenwert zwischen +1 und +32 767 verwendet werden.

Der Zählvorgang wird durch jedes Ansteuern mit einem „1“-Signal aktiviert. Der Zähleristwert wird dabei jeweils um den Wert 1 erhöht (Aufwärtszähler). Nach Erreichen des vorher festgelegten Zählersollwertes schaltet der Zähler auf den Signalzustand „1“.

HINWEIS

Die Angabe des Zählersollwertes kann auch indirekt über den in einem Datenregister gespeicherten dezimalen Zahlenwert vorgenommen werden.

Beispiel ▾

Einsatz der 16-Bit-Counter mit direkter Zählersollwertvorgabe

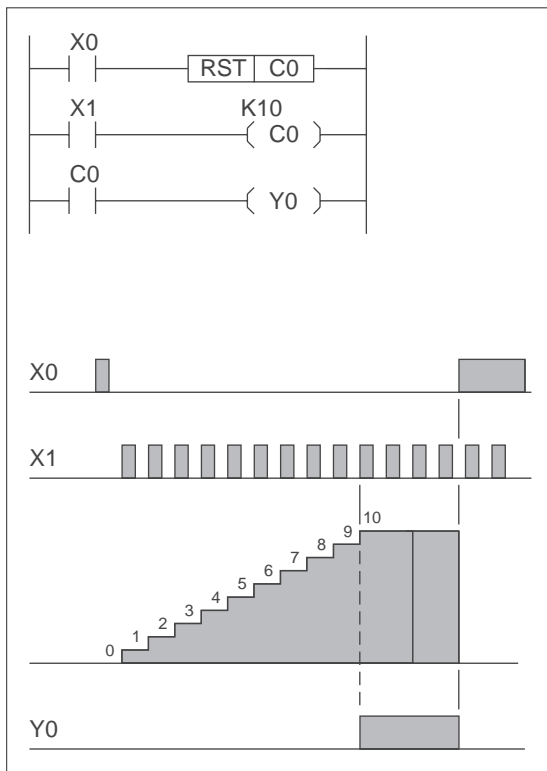


Abb. 3-9:
 Programmierbeispiel zum Einsatz der 16-Bit-Counter mit direkter Zählersollwertvorgabe

C000008C

Bei jedem Ansteuern mit einem „1“-Signal am Eingang X1 zählt der Counter C0 um den Zahlenwert 1 aufwärts. Der Ausgang Y0 wird nach 10 Zählsignalen am Eingang X1 gesetzt (Zählersollwert K10).

Nach Erreichen des Zählersollwertes K10 bleibt der Counter von den dann folgenden Setzimpulsen am Eingang X1 unbeeinflusst.

Über den Eingang X0 wird der Zähler mit Hilfe der RST-Anweisung zurückgesetzt. Der Istwert des Counters wird auf 0 gesetzt. Der Ausgang Y0 wird ausgeschaltet. △

Beispiel ▾

Einsatz der 16-Bit-Counter mit indirekter Zählersollwertvorgabe

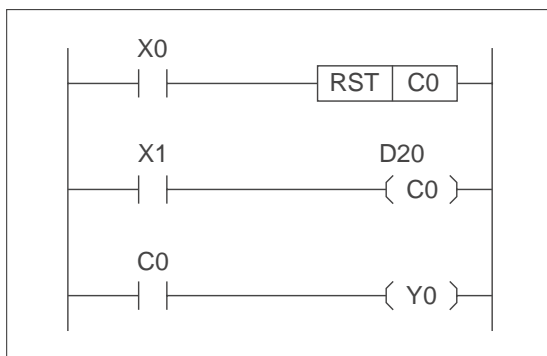


Abb. 3-10:
 Programmierbeispiel zum Einsatz der 16-Bit-Counter mit indirekter Zählersollwertvorgabe

C000028C

Der Zählersollwert wird indirekt über das Datenregister D20 festgelegt. △

3.5.2 32-Bit-Counter

32-Bit-Counter adressieren

Steuerung	Operandenadresse	Anzahl	Spannungsausfallsicher
FX1N	C200 bis C219	20	—
	C220 bis C234	15	●
FX2N	C200 bis C219	20	—
	C220 bis C234	15	●

Tab. 3-7: 32-Bit-Counter und die zugehörigen Operandenadressen

Die Counter C200 bis C234 sind Auf-/Abwärtszähler; die Zählrichtung wird durch den Zustand eines zugeordneten Sondermerkers vorgegeben. Die Zählrichtung kann während des Zählvorgangs geändert werden.

Zugeordnete Sondermerker

Operandenadresse	C200	C201	C202	C203	C204	C205	C206
Sondermerker	M8200	M8201	M8202	M8203	M8204	M8205	M8206
Operandenadresse	C207	C208	C209	C210	C211	C212	C213
Sondermerker	M8207	M8208	M8209	M8210	M8211	M8212	M8213
Operandenadresse	C214	C215	C216	C217	C218	C219	C220
Sondermerker	M8214	M8215	M8216	M8217	M8218	M8219	M8220
Operandenadresse	C221	C222	C223	C224	C225	C226	C227
Sondermerker	M8221	M8222	M8223	M8224	M8225	M8226	M8227
Operandenadresse	C228	C229	C230	C231	C232	C233	C234
Sondermerker	M8228	M8229	M8230	M8231	M8232	M8233	M8234

Tab. 3-8: Zuordnung der Sondermerker

Die Zählrichtung der Zähler wird durch den logischen Zustand des zugeordneten Sondermerkers festgelegt.

- Sondermerker eingeschaltet: Abwärtszähler
- Sondermerker ausgeschaltet: Aufwärtszähler

Die Arbeitsweise entspricht der eines 16-Bit-Zählers.

HINWEIS

Bei indirekter Adressierung werden 2 Datenregister benötigt. Zur Zuweisung von Sollwerten müssen 32-Bit-Anweisungen verwendet werden.

Beispiel ▾

Einsatz der 32-Bit-Counter mit direkter Zählervorgabe

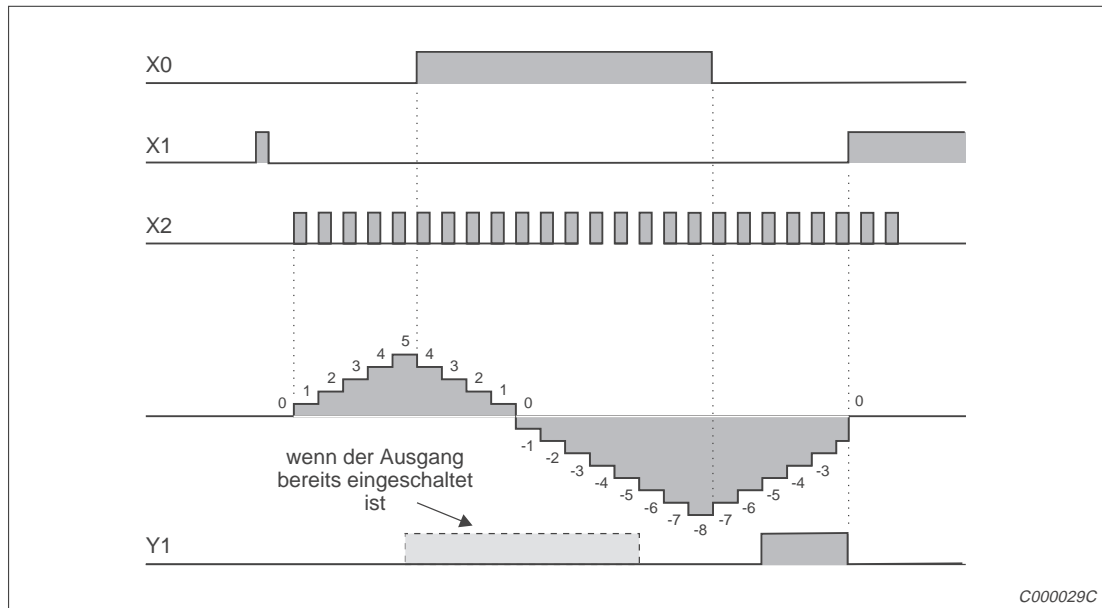


Abb. 3-11: Programmierbeispiel zum Einsatz der 32-Bit-Counter bei direkter Zählervorgabe

Sobald der Eingang X2 einschaltet, beginnt der Zählvorgang. Der Zähler C200 zählt die Einschaltimpulse von X2.

Der Ausgang Y1 wird eingeschaltet, wenn der Istwert von -6 auf -5 springt. Y1 wird zurückgesetzt, wenn der Istwert von -5 auf -6 springt.

Der Zählvorgang (aufwärts und abwärts) findet unabhängig vom aktuellen Status des Ausgangs statt. Wenn der Counter jedoch oberhalb von +2147483647 arbeitet, wird automatisch der Wert -2147483648 gültig. Wird unterhalb von -2147483648 gezählt, wird der Wert +2147483647 gültig.

Diese Counter werden „Ring-Counter“ genannt.

Über den Eingang X1 wird die RST-Anweisung ausgeführt. Der Istwert des Counters wird auf 0 gesetzt. Der Ausgang Y1 wird ausgeschaltet. △

Beispiel ▾

Einsatz der 32-Bit-Counter mit indirekter Zählervorgabe

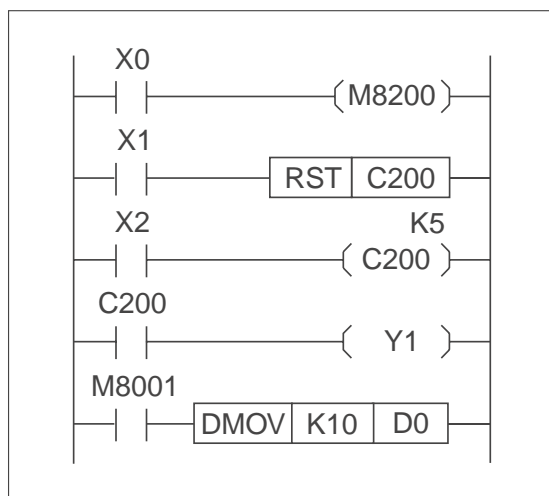


Abb. 3-12: Programmierbeispiel zum Einsatz der 32-Bit-Counter bei indirekter Zählervorgabe

C000030C

△

3.5.3 32-Bit-High-Speed-Counter

Die High-Speed-Counter sind 32-Bit-Counter, die schnelle externe Zählsignale verarbeiten. Als Zählengänge stehen die Eingänge X0 bis X7 zur Verfügung. Da eine Doppelbelegung der Eingänge nicht erlaubt ist, stehen pro SPS-Programm maximal sechs High-Speed-Counter zur Verfügung.

Den High-Speed-Countern sowie den verschiedenen Eingängen sind festgelegte Funktionen zugeordnet.

High-Speed-Counter arbeiten nach dem Prinzip des Interrupts. Der Vorteil hierbei ist, dass das Zählsignal unabhängig von der Programmzykluszeit verarbeitet wird.

Zuordnung der Eingänge

X	1-Phasen-Counter ohne Start und Reset						1-Phasen-Counter mit Start und Reset					2-Phasen-Counter Bidirektional					A/B-Phasen-Counter				
	C 235	C 236	C 237	C 238	C 239	C 240	C 241	C 242	C 243	C 244	C 245	C 246	C 247	C 248	C 249	C 250	C 251	C 252	C 253	C 254	C 255
X0	U/D						U/D			U/D		U	U		U		A	A		A	
X1		U/D					R			R		D	D		D		B	B		B	
X2			U/D					U/D			U/D		R		R			R		R	
X3				U/D				R			R			U		U			A		A
X4					U/D				U/D					D		D			B		B
X5						U/D			R					R		R			R		R
X6										S						S					S
X7											S					S					S

Tab. 3-12: High-Speed-Counter und die zugehörigen Zählengänge

U: Aufwärtszählender Eingang
 D: Abwärtszählender Eingang
 A: A-Phasen-Eingang
 B: B-Phasen-Eingang
 R: Reset-Eingang
 S: Start-Eingang

Die Eingänge X6 und X7 arbeiten nur als Startsignale. Sie können nicht für den High-Speed-Zählvorgang verwendet werden.

Es können mehrere High-Speed-Counter gleichzeitig im SPS-Programm eingesetzt werden.

Beim Einsatz verschiedener High-Speed-Counter muß darauf geachtet werden, daß kein Counter verwendet wird, dessen Eingänge bereits durch einen anderen Counter belegt sind. Eine Doppelbelegung von Eingängen ist nicht erlaubt.

Maximale Zählfrequenz und Zählgeschwindigkeit

Die maximale Frequenz bzw. Geschwindigkeit der Zählsignale, die noch von der SPS verarbeitet werden, beträgt beim Einsatz von nur einem High-Speed-Counter in einem SPS-Programm:

- 1- und 2-Phasen-Counter: max. 10 kHz
- AB-Phasen-Counter: max. 5 kHz

Die Summe aller Zählfrequenzen darf bei einer Steuerung der FX1S- oder FX1N-Serie 60 kHz und bei einer Steuerung aus der FX2N-Serie 30 kHz nicht überschreiten. Dabei ist zu beachten, dass die Frequenz von AB-Phasen-Countern in diese Berechnung mit dem doppelten Wert eingeht.

Die Eingänge X0 und X1 einer Steuerung der FX2N-Serie sind durch ihren Aufbau in der Lage, sehr hohe Frequenzen zu zählen:

Bei Verwendung der 1-Phasen-Counter C235, C236 oder C246 kann bis 60 kHz gezählt werden.

Bei Verwendung des 2-Phasen-Counters C251 kann bis 30 kHz gezählt werden.

Wenn Operationen zum Setzen- oder Rücksetzen von High-Speed-Countern (Anweisungen DHSCS, FNC 53 und DHSCR, FNC54) oder ein Bereichsvergleich (DHSZ, FNC55) im Programm verwendet werden, müssen die Eingänge X0 und X1 als Zählwege benutzt werden. In diesem Fall gilt die folgende Tabelle.

Steuerung	Anweisung	Summe aller Zälfrequenzen
FX1S FX1N	FNC53 oder FNC54	30 kHz
FX2N	FNC53 oder FNC54	11 kHz
	FNC55	5,5 kHz

Tab. 3-13:

Einschränkungen bei Verwendung der Anweisungen FNC53, FNC54 und FNC55

Die Istwerte aller High-Speed-Counter werden in einem spannungsausfallsicheren Speicher abgelegt.

HINWEISE

Die Zählwege X0 bis X5 dürfen nicht als Einschaltbedingungen für High-Speed-Counter programmiert werden.

High-Speed-Counter können nicht im Zusammenhang mit 16-Bit-Anweisungen eingesetzt werden.

Die Summe aller Zälfrequenzen an allen SPS-Eingängen muß ≤ 60 kHz sein (AB-Phasen-Counter sind doppelt zu zählen).

Die SPD-Anweisung (FNC 56) hat die Counter- und Interrupt-Charakteristik eines High-Speed-Counters. Aus diesem Grund sollten für die SPD-Anweisung die Eingänge X0 bis X5 verwendet werden. Auch für diese Eingänge gilt, daß sie nicht gleichzeitig von anderen High-Speed-Countern verwendet werden können.

1-Phasen-Counter mit einem Zähleringang

1-Phasen-Counter sind High-Speed-Counter mit nur einem Zähleringang.

Die 1-Phasen-Counter lassen sich in drei Gruppen einteilen:

- Ohne Start- und Reset-Eingang (C235 bis C240)
- Mit Reset-Eingang (C241 bis C243)
- Mit Start- und Reset-Eingang (C244 bis C245)

Die Zählrichtung (auf- oder abwärtszählend) wird durch Einschalten eines Sondermerkers festgelegt.

Sondermerker eingeschaltet: Abwärtszählend

Sondermerker ausgeschaltet: Aufwärtszählend

1-Phasen-Counter	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245
Sondermerker	M8235	M8236	M8237	M8238	M8239	M8240	M8241	M8242	M8243	M8244	M8245

Tab. 3-11: 1-Phasen-Counter und die zugehörigen Sondermerker

Der Zählbereich geht von -2 147 483 648 bis 2 147 483 647.

Beispiel ▾ Einsatz eines 1-Phasen-Counters mit Start- und Reset-Eingang (C244).

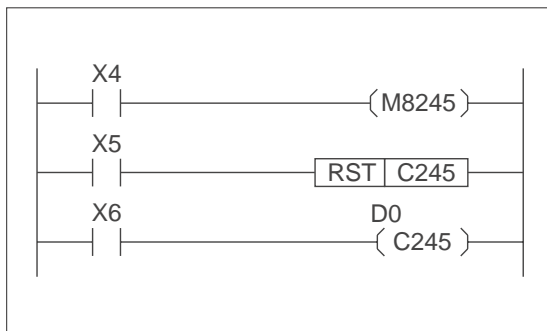


Abb. 3-14:

Programmierbeispiel zum Einsatz eines 1-Phasen-Counters mit Start- und Reset-Eingang (C244)

C000123C

Wenn der Sondermerker M8245 eingeschaltet ist, zählt der Zähler M8245 abwärts. Ist der Merker M 8245 nicht gesetzt, zählt der Zähler aufwärts. Mit dem Schalter X5 wird der Istwert des Zählers auf den Wert 0 zurückgesetzt. Dies ist ebenfalls über den automatisch zugeordneten Rücksetzeingang X3 möglich.

Durch das Einschalten von X6 und dem automatisch zugeordneten Starteingang X7 wird der Zähler aktiviert und zählt die Impulse aus seinem Zählengang X2. Da es sich um einen 32-Bit-Zähler handelt, werden die Register D0 und D1 für die Sollwertvorgabe genutzt.

Beispiel ▾ Im Vergleich zu den im Programm verwendeten Eingängen X5 und X6 bietet die Benutzung der Eingänge X7 und X3 den Vorteil, daß die Verarbeitung der externen Start- und Reset-Signale nicht von der Programmzykluszeit abhängig ist.

△

2-Phasen-Counter mit zwei Zählwegen

2-Phasen-Counter verfügen über je einen Zählweg zum Auf- und Abwärtszählen.

Die 2-Phasen-Counter lassen sich in drei Gruppen einteilen:

- Ohne Start- und Reset-Eingang (C246)
- Mit Reset-Eingang (C247, C248)
- Mit Start- und Reset-Eingang (C249, C250)

Die SPS setzt automatisch einen Sondermerker, der die aktuelle Zählrichtung des 2-Phasen-Counters anzeigt:

Sondermerker eingeschaltet: Abwärtszählend

Sondermerker ausgeschaltet: Aufwärtszählend

2-Phasen-Counter	C246	C247	C248	C249	C250
Sondermerker	M8246	M8247	M8248	M8249	M8250

Tab. 3-12:
2-Phasen-Counter und zugehörige Sondermerker

Der Zählbereich geht von -2 147 483 648 bis 2 147 483 647.

Beispiel ▾

Einsatz eines 2-Phasen-Counters ohne Start- und Reset-Eingang (C246).

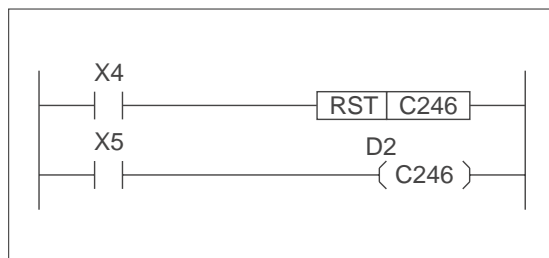


Abb. 3-15:
Programmierbeispiel zum Einsatz eines 2-Phasen-Counters ohne Start- und Reset-Eingang (C246)

C000124C

Wird der Eingang X4 eingeschaltet, wird der Counter C246 zurückgesetzt. Der Zählvorgang wird gestartet, wenn der Eingang X5 eingeschaltet ist. Für den Counter C246 sind die Zählwege X0 und X1 zum Aufwärts- und Abwärtszählen reserviert.

Bei Signalimpulsen am Eingang X0 zählt der Counter aufwärts, und bei Signalimpulsen am Eingang X1 zählt der Counter abwärts. △

AB-Phasen-Counter mit zwei Zählengängen

Die AB-Phasen-Counter verfügen über je einen A- und einen B-Phasen-Zählengang. Über die Signale an den A- und B-Phasen-Eingängen wird festgelegt, ob der Counter aufwärts oder abwärts zählen soll.

- Aufwärtszählend

A-Phasen-Eingang: „1“-Signal

B-Phasen-Eingang: ansteigende Signalfanke (Signalwechsel von „0“ auf „1“)

- Abwärtszählend

A-Phasen-Eingang: „1“-Signal

B-Phasen-Eingang: abfallende Signalfanke (Signalwechsel von „1“ auf „0“)

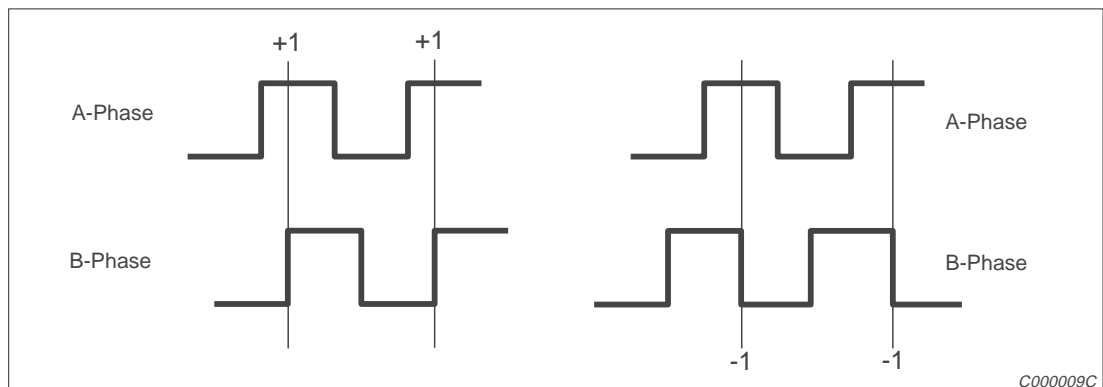


Abb. 3-16: AB-Phasen-Counter

Jeder Eingangssignalwechsel mit ansteigender Flanke am B-Phasen-Eingang läßt den Zähler um 1 aufwärtszählen und jeder Signalwechsel mit abfallender Flanke um 1 abwärtszählen. Während der Signalwechsel muß am A-Phasen-Eingang ein „1“-Signal anliegen.

Die AB-Phasen-Counter lassen in sich drei Gruppen einteilen:

- Ohne Start- und Reset-Eingang (C251)
- Mit Reset-Eingang (C252, C253)
- Mit Start- und Reset-Eingang (C254, C255)

Die SPS setzt automatisch einen Sondermerker, der die aktuelle Zählrichtung des 2-Phasen-Counters anzeigt:

Sondermerker eingeschaltet: Abwärtszählend

Sondermerker ausgeschaltet: Aufwärtszählend

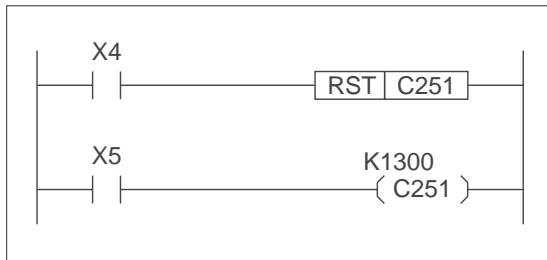
AB-Phasen-Counter	C251	C252	C253	C254	C255
Sondermerker	M8251	M8252	M8253	M8254	M8255

Tab. 3-13:
AB-Phasen-Counter und zugehörige Sondermerker

Der Zählbereich geht von -2 147 483 648 bis 2 147 483 647.

Beispiel ▾

Einsatz eines AB-Phasen-Counters ohne Start- und Reset-Eingang (C251)

**Abb. 3-17:**

Programmierbeispiel zum Einsatz eines AB-Phasen-Counters ohne Start- und Reset-Eingang (C251)

C000126C

Bei eingeschaltetem Eingang X5 zählt der Counter C251 die Signale an den Zählengängen X0 (A-Phasen-Eingang) und X1 (B-Phasen-Eingang). △

3.6 Schrittstatus

Schrittstatusoperanden werden in Zusammenhang mit Ablaufsteuerungen (STL-Anweisung) eingesetzt. Mit den Schrittstatusoperanden werden die einzelnen Schritte einer Ablaufsteuerung festgelegt.

3.6.1 Schrittstatusoperanden adressieren

Es stehen bis zu 1000 Schrittstatusoperanden im Bereich von S0 bis S999 zur Verfügung.

Die Schrittstatusoperanden S lassen sich in zwei Gruppen einteilen:

- S0 bis S9 (10 Adressen): Schrittstatus initialisieren
- S10 bis S999: frei wählbarer Einsatzbereich

Steuerung	Operanden	davon gepuffert
FX1S	S0 – S127 (128)	S0 – S127 (128)
FX1N	S0 – S999 (1000)	S0 – S999 (1000)
FX2N	S0 – S999 (1000)	S500 – S999 (500)

Tab. 3-14:
Übersicht der
Schrittstatusoperanden

Detaillierte Informationen zum Einsatz der STL-Anweisung und der Schrittstatusoperanden S enthält Abschnitt 5.1.

Werden in einem Programm keine Schrittsteuerungen verwendet, können die Schrittstatusoperanden S wie Merker eingesetzt werden.

Beispiel ▾

Einsatz der Schrittstatusoperanden

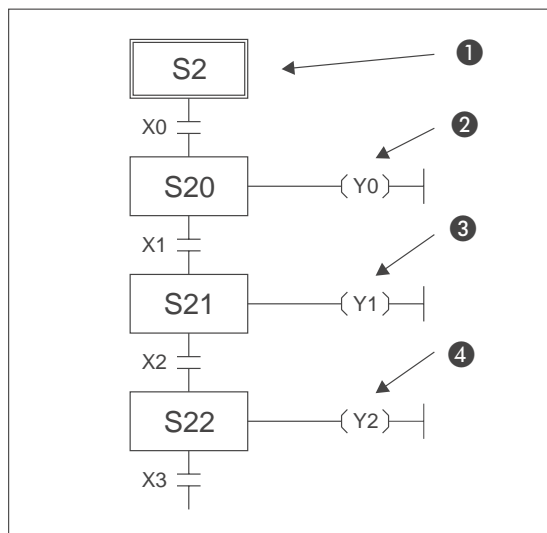


Abb. 3-18:
Programmierbeispiel zum Einsatz der
Schrittstatusoperanden

- ① Initialisierungsstatus
- ② Absenken
- ③ Greifen
- ④ Anheben

C000127C

Der Schrittstatusoperand S20 wird aktiviert, wenn der Eingang X0 eingeschaltet wird. Anschließend wird der Ausgang Y0 (② Absenken) eingeschaltet.

Nach Erreichen des unteren Endschalters X1 wird der Schrittstatusoperand S21 aktiviert, indem der Ausgang Y1 (③ Greifen) eingeschaltet wird.

Wird der Endschalter X2 erreicht, wird S22 aktiviert. Der Ausgang Y2 (④ Anheben) wird eingeschaltet. △

3.7 Dezimal-, Hexadezimalkonstanten

Mit den Dezimal- und Hexadezimalkonstanten (K,H) lassen sich numerische Zahlenwerte innerhalb eines SPS-Programms festlegen (z. B. Zeit-, Zählersollwert). Der Zahlenwert wird intern von der SPS in einen binären Zahlenwert codiert.

Im Abs. 3.8.7 sind alle wichtigen Zahlensysteme und deren Codierungen untereinander ausführlich beschrieben.

3.7.1 Zahlenwertebereiche der Dezimal-, Hexadezimalkonstanten

Konstanten	16 Bit	32 Bit
dezimal K	-32 768 bis +32 767	-2 147 483 648 bis +2 147 483 647
hexadezimal H	0 bis FFFF	0 bis FFFFFFFF

Tab. 3-15: Zahlenwertebereiche der Dezimal-, Hexadezimalkonstanten

3.8 Register

Register stellen einen Datenspeicher innerhalb der SPS dar. In einem Register können Sie Zahlenwerte und aufeinanderfolgende binäre Informationen zusammenfassen und abspeichern. Dadurch lassen sich z. B. die Signalzustände mehrerer Eingänge auf einmal abspeichern und im Programm verarbeiten.

Die Daten werden in einem 16-Bit-Format abgespeichert. Durch Zusammenschalten von zwei 16-Bit-Registern haben Sie die Möglichkeit, „Doppelregister“ zu bilden. In einem Doppelregister können Daten in einem 32-Bit-Format abgespeichert werden.

3.8.1 Einteilung der Register

Folgende Registertypen stehen zur Verfügung:

- **Datenregister** (ungepuffert)
Register ohne Datensicherung bei einem Spannungsausfall der SPS.
- **Datenregister** (gepuffert)
Register mit Datensicherung bei einem Spannungsausfall der SPS. Die Daten werden in einem spannungsausfallsicheren Speicher abgespeichert.
- **Indexregister**
Diese Register dienen zum Speichern von Zwischenergebnissen und zur Indizierung von Operanden. Nähere Angaben siehe Abs. 3.8.5.
- **Sonderregister**
Für bestimmte Kontroll- oder Überwachungsfunktionen stehen eine Reihe von speziellen Registern zur Verfügung. Nähere Angaben siehe Abs. 3.8.4.
- **File-Register**
Zum Speichern von Parametern oder Rezepturen werden File-Register benötigt. Bei einer Steuerung der FX1N- oder der FX2N-Serie werden die Speicherbereiche dieser Register vom Anwender festgelegt. Diese File-Register sind ein Teil der Latch-File-Register.

3.8.2 Aufbau der Register

Jedes Register besteht aus einem Vorzeichenbit und mehreren Datenbits.

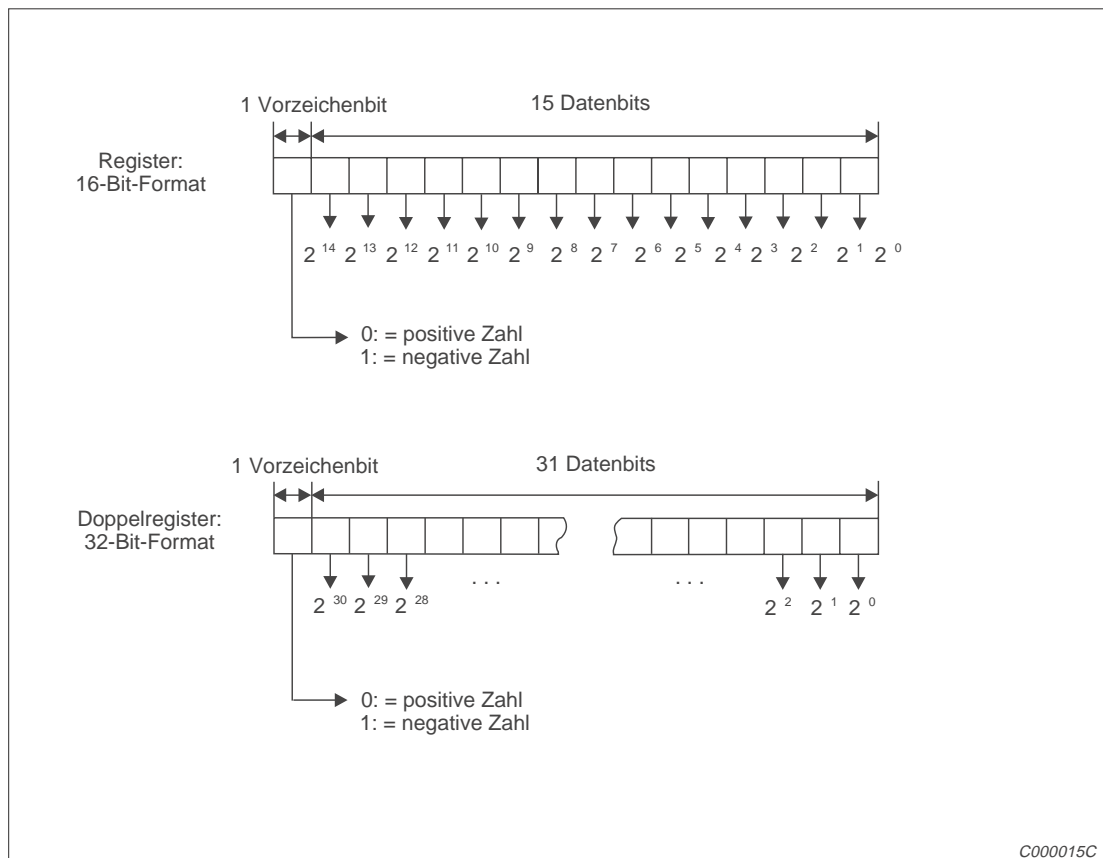


Abb. 3-19: Aufbau der Register (16 Bit) und der Doppelregister (32 Bit)

3.8.3 Adressierung der Register

Die Adressierung der Datenregister erfolgt dezimal. Bei einem Doppelregister beginnt die Adressierung mit dem unteren 16-Bit-Register.

Steuerung	Register	Adressen	Anzahl	davon gepuffert	Anzahl
FX1S	Datenregister	D0 – D255	256	D128 – D255	128
	Indexregister	V, Z	16	—	—
	Sonderregister	D8000 – D8255	256	D8000 – D8255	256
	Extern veränderbare Register ^①	D8030 und D8031, Werte von 0 bis 255 sind durch integrierte Potentiometer einstellbar	2	—	—
	File-Register	D1000 – D2499	1500	D1000 – D2499	1500
FX1N	Datenregister	D0 – D7999	8000	D128 – D7999	7872
	Indexregister	V, Z	16	—	—
	Sonderregister	D8000 – D8255	256	D8000 – D8255	256
	Extern veränderbare Register ^①	D8030 und D8031, Werte von 0 bis 255 sind durch integrierte Potentiometer einstellbar	2	—	—
	File-Register	D1000 – D7999	7000 (anteilig)	D1000 – D7999	7000 (anteilig)
FX2N	Datenregister	D0 – D7999	8000	D200 – D7999*	7800
	Indexregister	V0 – V7, Z0 – Z7	16	—	—
	Sonderregister	D8000 – D8255	256	D8000 – D8255	256
	File-Register	D1000 – D7999	7000 (anteilig)	D1000 – D7999	7000 (anteilig)

Tab. 3-16: Register und die zugehörigen Operandenadressen

① Diese beiden Register sind auch bei der Anzahl der Sonderregister enthalten.

HINWEIS

Von dem angegebenen Datenregisterbereich der FX2N-Serie kann durch den Anwender nur bei den Registern D200 bis D511 bestimmt werden, ob diese gepuffert sind oder nicht. Die Datenregister ab D512 sind in der FX2N immer gepuffert.

3.8.4 Verwendung der Sonderregister

Den Sonderregistern D8000 bis D8255 sind intern feste Kontroll- oder Überwachungsfunktionen (Monitor-Funktionen) zugeordnet.

Beim Einschalten der Steuerung werden von der Systemsoftware automatisch die Standardwerte in die Sonderregister geschrieben. So wird z. B. automatisch der Datenwert des Watch Dog Timers in das Sonderregister D8000 geschrieben. Möchten Sie diesen Datenwert verändern, dann müssen Sie den alten Datenwert mit Hilfe einer MOV-Anweisung überschreiben (weitere Hinweise siehe Abs. 6.3.3).

HINWEISE

Die Daten gehen beim Schalten der Steuerung in den STOP-Modus nicht verloren. Ein Spannungsausfall führt jedoch zu einem Datenverlust.

Es dürfen nur belegte Datenregister eingesetzt werden.

In Kapitel 9 sind alle vorhandenen Sonderregister und deren Funktionen aufgeführt.

3.8.5 Extern veränderbare Register

Bei den Steuerungen der FX1S- und der FX1N-Serie sind zwei Potentiometer integriert, mit denen der Inhalt zweier Sonderdatenregister im Bereich von 0 bis 255 verändert werden kann. Für Steuerungen der FX2N-Serie ist der Sollwertvorgabe-Adapter FX2N-8AV-BD erhältlich, mit dem die Eingabe von 8 analogen Sollwerten möglich ist. Das Einlesen der Sollwerte sowie die Bereichseinteilung der Potentiometer erfolgt durch die Applikationsanweisungen VRRD (FNC85) und VRSC (FNC86).

Mit Hilfe der Potentiometer können z. B. Sollwerte für Timer und Counter verändert werden, ohne ein Programmiergerät anschließen zu müssen.

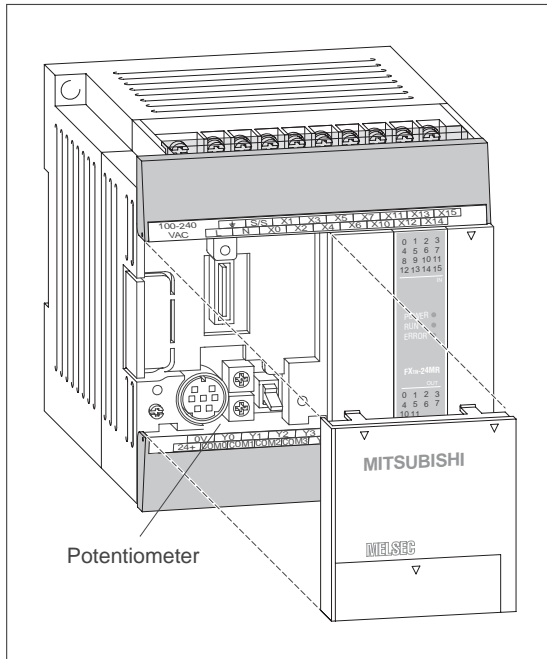


Abb. 3-20:
Potentiometer an einer Steuerung der FX1S/FX1N-Serie

C000413C

	FX1S	FX1N	FX2N
Anzahl der Potentiometer	2 integrierte Potentiometer (VR1 und VR2), bei Verwendung des Sollwertvorgabe-Adapters FX2N-8AV-BD können 8 zusätzliche Potentiometer genutzt werden		Bei Verwendung des Sollwertvorgabe-Adapters FX2N-8AV-BD können 8 Potentiometer genutzt werden
Extern veränderbare Register	D8030 (wird von VR1) beeinflusst D8031 (wird von VR1) beeinflusst Die zusätzlichen 8 Register bei Verwendung des Sollwertvorgabe-Adapters FX2N-8AV-BD werden durch den Anwender bestimmt.		Die 8 Register werden durch den Anwender mit den Anweisungen VRRD und VRSC festgelegt.

Tab. 3-17: Extern veränderbare Register

3.8.6 Einsatz der Indexregister

Die Indexregister werden verwendet, um bei Transfer- und Vergleichsanweisungen zur Operandenadresse einen Indexwert zu addieren.

Die Indexregister sind 16-Bit-Register.

In 32-Bit-Anweisungen können die Indexregister V (V0 – V7) und Z (Z0 – Z7) kombiniert eingesetzt werden. Z speichert die unteren 16 Bit, und V speichert die oberen 16 Bit. Als Zieladresse ist das Indexregister Z anzugeben. Indexregister selbst können nicht indiziert werden.

Beispiel ▾

Datentransfer vom Datenregister D5V zum Datenregister D10Z

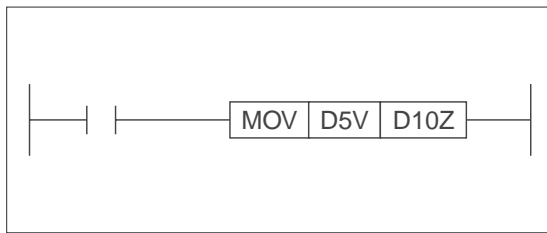


Abb. 3-21:

Programmierbeispiel zum Datentransfer vom Datenregister D5V zum Datenregister D10Z

C000044C

Berechnung der Ausgangsadresse D5V:

$$V = 8$$

$$5 + 8 = 13 \rightarrow D13$$

Berechnung der Zieladresse D10Z:

$$Z = 14$$

$$10 + 14 = 24 \rightarrow D24$$

Es findet demnach ein Datentransfer vom Datenregister D13 zum Datenregister D24 statt.

△

3.8.7 Einsatz der File-Register

File-Register werden in Blöcken von 500 Adressen im Programmspeicherbereich (EPROM oder EEPROM) der Steuerung abgelegt. Die Anzahl der Blöcke wird über die Parameter festgelegt. Der Zugang zu den File-Registern ist über Programmiergeräte und Bedienterminals möglich.

HINWEISE

Wenn File-Register eingesetzt werden, verringert sich der für das SPS-Programm nutzbare Speicherbereich. Für jeden Block von 500 File-Registern verringert sich die Anzahl der nutzbaren Programmschritte um 500 Schritte. Die Anzahl der File-Register variiert in Abhängigkeit vom Steuerungstyp.

Bei der Verwendung der File-Register der FX1N- und der FX2N-Serie ist zu beachten, daß der Speicherbereich mit dem gepufferten Speicherbereich überlappt.

Lesen von File-Registern

Während des Betriebes der SPS können die Daten der File-Register mittels der BMOV-Anweisung ausgelesen werden.

Schreiben von File-Registern

Bei der FX1S-Serie können die File-Register nur mit Programmiergeräten oder mit PCs und entsprechender Software beschrieben werden.

Bei Steuerungen der FX1N- und der FX2N-Serie können die Inhalte der File-Register auch vom SPS-Programm unter Verwendung der BMOV-Anweisung verändert werden.

Für nähere Informationen verwenden Sie bitte die Anleitungen der entsprechenden Programmiersysteme.

Bei der FX1S ist eine Veränderung der Daten im RUN-Modus nicht möglich.

HINWEISE

Eine Veränderung der File-Registerdaten ist im RUN-Modus nur bei RAM-Registern oder File-Registern im internen Speicher möglich.

File-Register, die sich im RAM, im internen Speicher oder auf EEPROM-Speicherkassetten befinden, können im STOP-Modus geändert werden.

File-Register, die sich auf einer EPROM-Speicherkassette befinden, können nicht verändert werden.

Zahlenwertbereiche von Datenregistern

Werden binär codierte Zahlen in einem Register abgespeichert, ist der Zahlenwertebereich aufgrund der begrenzten Größe eines Registers eingeschränkt.

- Dezimalzahlen
16 Bit: -32 768 bis +32 767 32 Bit: -2 147 483 648 bis +2 147 483 647
- Hexadezimalzahlen
16 Bit: 0 bis FFFF 32 Bit: 0 bis FFFFFFFF

Darstellung negativer Zahlen

Negative Zahlen werden als 2er-Komplement dargestellt.

Bei der Bildung eines 2er-Komplements wird die duale Zahl invertiert (1er-Komplementbildung) und anschließend der binäre Zahlenwert 1 addiert.

Beispiel ▾

0101101 (dual) → +45 (dezimal)
 1010010 (dual) → 1er-Komplement
 1010011 (dual) → 2er-Komplement
 1010011 (dual) → -45 (dezimal)

△

Der im Datenregister abgelegte Wert ist negativ, wenn im höchstwertigen Bit (Vorzeichenbit) die Zahl 1 steht.

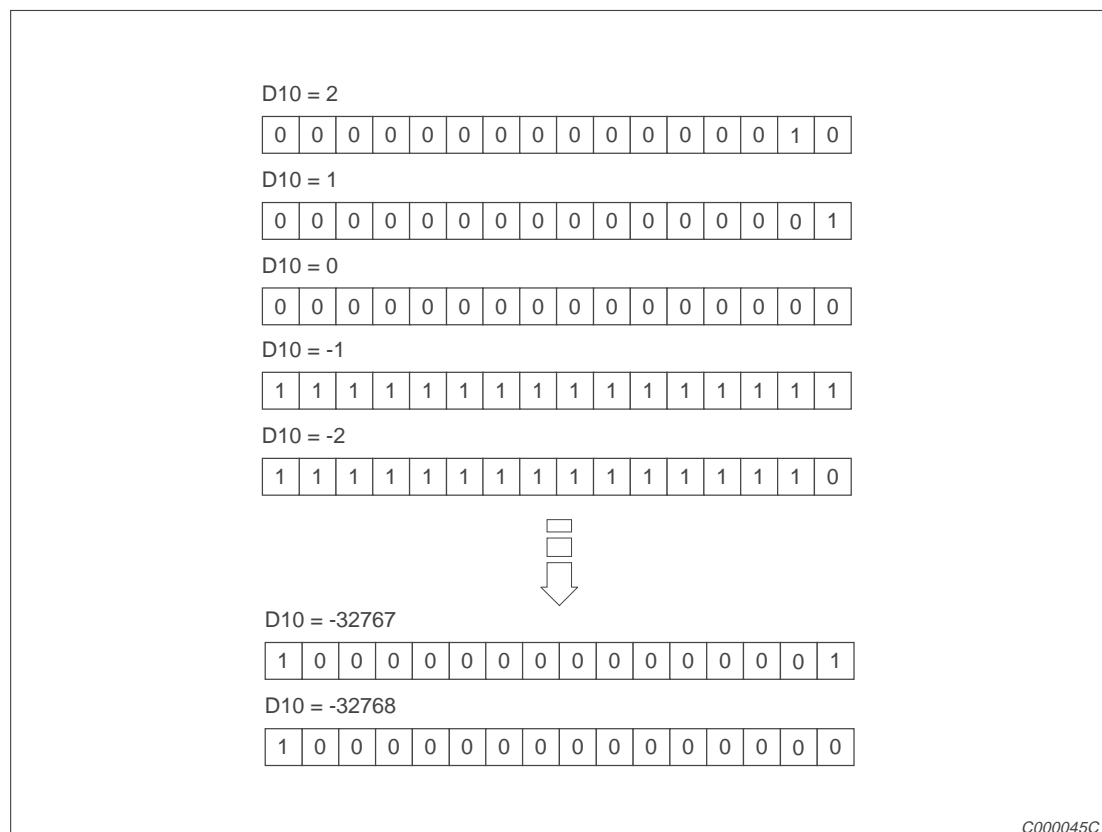


Abb. 3-22: Beispiel für die Darstellung negativer Zahlen

△

3.8.8 Zahlendarstellungen

Die Steuerungen der FX-Familie geben Ihnen die Möglichkeit, mit Zahlenwerten in folgenden Darstellungen zu arbeiten:

- Dezimalzahlen
- Zahlen im wissenschaftlichen Format
- Fließkomma-Zahlen
- Dualzahlen (Binärzahlen)
- Hexadezimalzahlen
- BCD-Format
- Bit-Muster

Interne Darstellungen der Zahlen in der SPS

Der Mikroprozessor der SPS verarbeitet grundsätzlich nur binäre Informationen. Die kleinste Einheit einer binären Information heißt Bit. Mit einem Bit lassen sich die zwei Signalzustände „0“ und „1“ darstellen.

Alle Zahlenwerte, die nicht in einem binären Format vorliegen, werden deshalb von der SPS in ein binäres Format umgewandelt (codiert).

HINWEIS

Die SPS stellt intern alle Zahlen als 16- oder 32-stellige Dualzahlen oder als Bit-Muster (16-Bit-, 32-Bit-Format) dar.

In den nächsten Abschnitten folgt eine Einführung in die verschiedenen Zahlensysteme und die Umwandlung von Zahlenwerten (Codierungen) zwischen diesen Zahlensystemen.

Dezimalzahlensystem

Basis: 10

Ziffern: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Wertebereich:

- 16-Bit-Format: -32768 bis +32767
- 32-Bit-Format: -2147483648 bis +2147483647

Beispiel ▾

$$351 \text{ (dezimal)} = 3 \times 10^2 + 5 \times 10^1 + 1 \times 10^0$$



Zahlen im wissenschaftlichen Format

Dieses Format ist an die wissenschaftliche Darstellung von besonders grossen und besonders kleinen Zahlen angelehnt. Die Darstellung erfolgt im 32-Bit-Format mit Fließkomma.

Format: Mantisse x 10^{Exponent}

Wertebereich:

- Mantisse: ±1000 bis 9999, oder 0
- Exponent: -41 bis +35

Beispiel ▾

Geschwindigkeit des Lichtes:

- als Dezimalzahl: 299792458 m/s
- im wissenschaftlichen Format: 2998×10^5 m/s

Hierbei ist 2998 die Mantisse und 5 der Exponent. In Datenregistern gespeichert hätte die Zahl zum Beispiel die Form $D120 \times 10^{D121}$.

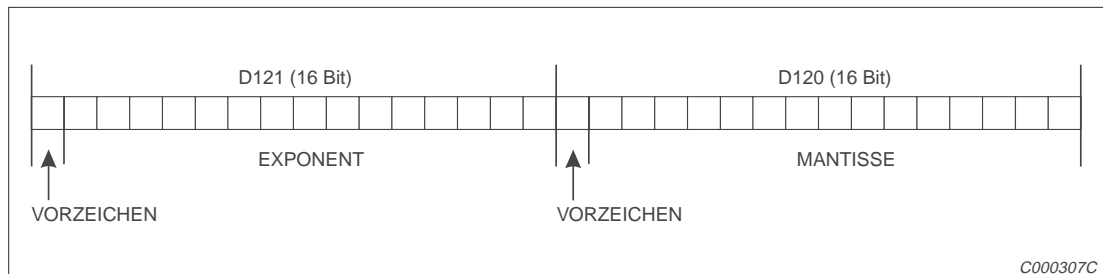


Abb. 3-23: Belegung im Datenregister

△

Fließkomma-Zahlensystem

Da Operationen mit Zahlen sehr schnell den zulässigen Wertebereich überschreiten würden, bietet die FX-Serie zusätzlich die Darstellung sehr grosser und sehr kleiner Zahlen im Fließkomma-Format, wie es in Personal- und Mikro-Computern eingesetzt wird.

Das Format des Fließkomma-Zahlensystems speichert Mantisse und Exponent als Binärzahlen in einem 32-Bit-Doppelwort, wobei die Mantisse 23 Bit und der Exponent 8 Bit belegt.

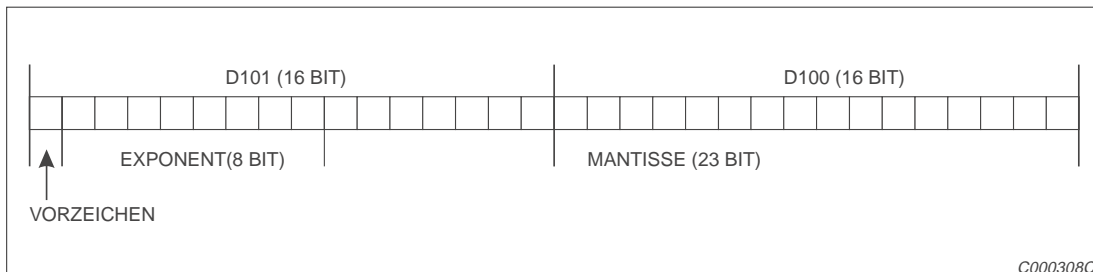


Abb. 3-24: Belegung im Datenregister

Format: $\pm \text{Mantisse} \times 2^{\text{Exponent}}$

Wertebereich:

$$\text{Mantisse: } 1 \times 2^0 + A_{22} \times 2^{-1} + A_{21} \times 2^{-2} + \dots + A_0 \times 2^{-23}$$

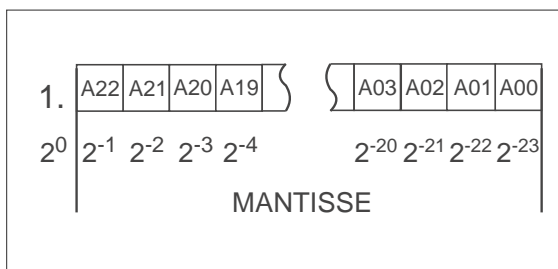


Abb. 3-25: Mantisse

C000309C

Exponent: $(E_7 \times 2^7 + E_6 \times 2^6 + \dots + E_0 \times 2^0) - 127$, das ergibt -126 bis +127

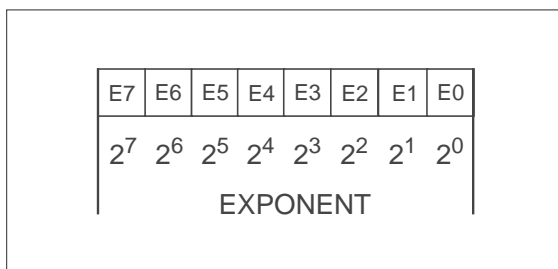


Abb. 3-26: Exponent

C000310C

Codierung: Dualzahl → Dezimalzahl**Beispiel ▾**

111000 (dual)

$$111000 \text{ (dual)} = 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$$

$$111000 \text{ (dual)} = 32 + 6 + 8$$

111000 (dual) = 56 (dezimal)

△

Oktalzahlsystem

Basis: 8

Ziffern: 0, 1, 2, 3, 4, 5, 6, 7

Beispiel ▾

245 (oktal)

$$245 \text{ (oktal)} = 2 \times 8^2 + 4 \times 8^1 + 5 \times 8^0$$

$$245 \text{ (oktal)} = 128 + 32 + 5$$

245 (oktal) = 165 (dezimal)

△

Codierung: Dezimalzahl → Oktalzahl**Beispiel ▾**

30 (dezimal)

$$30 : 8 = 3 \text{ Rest } 6$$

$$3 : 8 = 0 \text{ Rest } 3$$

30 (dezimal) = 36 (oktal)

△

Codierung: Oktalzahl → Dezimalzahl**Beispiel ▾**

374 (oktal)

$$374 \text{ (oktal)} = 3 \times 8^2 + 7 \times 8^1 + 4 \times 8^0$$

$$374 \text{ (oktal)} = 192 + 56 + 4$$

374 (oktal) = 252 (dezimal)

△

Hexadezimalzahlensystem

Basis: 16

Ziffern: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

(A=10, B=11, C=12, D=13, E=14, F=15)

Beispiel ▾

1E (hexadezimal)

$$1E \text{ (hexadezimal)} = 1 \times 16^1 + 14 \times 16^0$$

$$1E \text{ (hexadezimal)} = 16 + 14$$

$$1E \text{ (hexadezimal)} = 30 \text{ (dezimal)}$$

△

Codierung: Dezimalzahl → Hexadezimalzahl**Beispiel ▾**

63 (dezimal)

$$63 : 16 = 3 \text{ Rest } 15 \rightarrow F \text{ (hexadezimal)}$$

$$3 : 16 = 0 \text{ Rest } 3 \rightarrow 3 \text{ (hexadezimal)}$$

$$63 \text{ (dezimal)} = 3F \text{ (hexadezimal)}$$

△

Codierung: Hexadezimalzahl → Dezimalzahl**Beispiel ▾**

7A (hexadezimal)

$$7A \text{ (hexadezimal)} = 7 \times 16^1 + 10 \times 16^0$$

$$7A \text{ (hexadezimal)} = 112 + 10$$

$$7A \text{ (hexadezimal)} = 122 \text{ (dezimal)}$$

△

BCD-Format

Im BCD-Format (**Binär-Codierte-Dezimalziffer**) wird jede Ziffer einer Dezimalzahl durch eine 4-Bit-Binärzahl dargestellt. Bei einer 4-Bit-Darstellung ist es möglich, die dezimalen Ziffern 0 bis 15 binär zu codieren. Im BCD-Format ist jedoch nur die Codierung der dezimalen Ziffern von 0 bis 9 zulässig.

Codierung: Dezimalzahl → BCD-Format

Beispiel ▾

67 (dezimal) \Leftrightarrow Ziffern: 6, 7

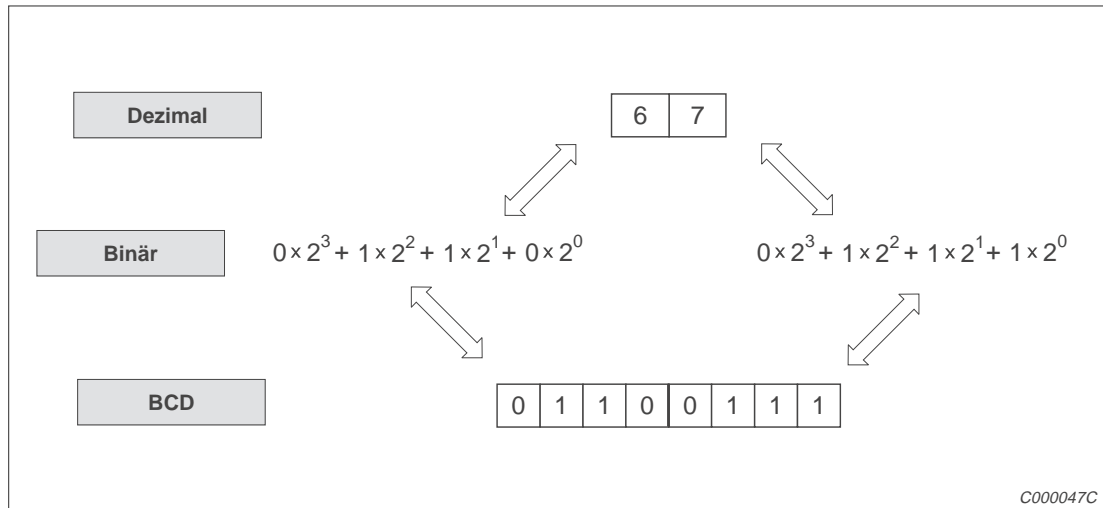


Abb. 3-28: Codierung einer Dezimalzahl in ein BCD-Format bzw. umgekehrt



3.9 Pointer

Pointer werden in Zusammenhang mit der CJ-Sprunganweisung oder der CALL-Anweisung programmiert.

Pointer sind Sprungzieladressen, mit denen das Sprungziel oder das Unterprogramm im Programm markiert werden (Pointer-Markierung).

3.9.1 Pointer adressieren

Bei einer Steuerung der FX1S-Serie stehen die Pointer-Markierungen P0 bis P63 (64 Adressen) zur Verfügung. Bei der FX1N- und FX2N-Serie umfasst der Pointerbereich 128 Adressen (P0 bis P127). In Verbindung mit der Anweisung CJ kann das Programmende angesprungen werden, wenn die Pointer-Adresse P63 verwendet wird. Die END-Anweisung braucht in diesem Fall keine Pointer-Markierung.

HINWEIS

Die gleiche Pointer-Markierung darf nicht mehrfach in einem SPS-Programm benutzt werden.

3.9.2 Nesting-Ebenen

Während der Ausführung eines Interrupts sind alle anderen Interrupts inaktiv. Um geschachtelte Interrupts zu erhalten müssen die EI-DI-Anweisungen innerhalb einer Interrupt-Routine programmiert werden. Die Interrupts können in zwei Nesting-Ebenen geschachtelt werden.

3.10 Interrupt-Pointer

Mit Hilfe der Interrupt-Pointer kann ein Sprung innerhalb eines SPS-Programms zu einem Interrupt-Programm durchgeführt werden (siehe auch Abs. 6.2.4).

3.10.1 Interrupt-Pointer adressieren

FX1S/FX1N MELSEC FX1S und FX1N:

Es stehen 6 Interrupt-Pointer zur Verfügung. Die Adressierung eines Interrupt-Pointers müssen Sie wie folgt vornehmen:

Interrupt-Pointer: I ① 0 ②

- ① Adresse 0 bis 3; entspricht Eingang X0 bis X3
- ② 0: = Interrupt bei abfallender Eingangssignalfanke
1: = Interrupt bei ansteigender Eingangssignalfanke

Beispiel ▾

Interrupt-Pointer: I201

Das über den Interrupt-Pointer aufgerufene Interrupt-Programm wird bei ansteigender Signalfanke am Eingang X2 ausgeführt.

Die Rückkehr ins Hauptprogramm erfolgt, nachdem die IRET-Anweisung ausgeführt wurde.

△

FX1S

Bei einer Steuerung der FX1S-Serie mit den Versionsnummern 1.00 bis 1.30 muss das Programm erweitert werden, wenn Interrupt-Pointer verwendet werden.

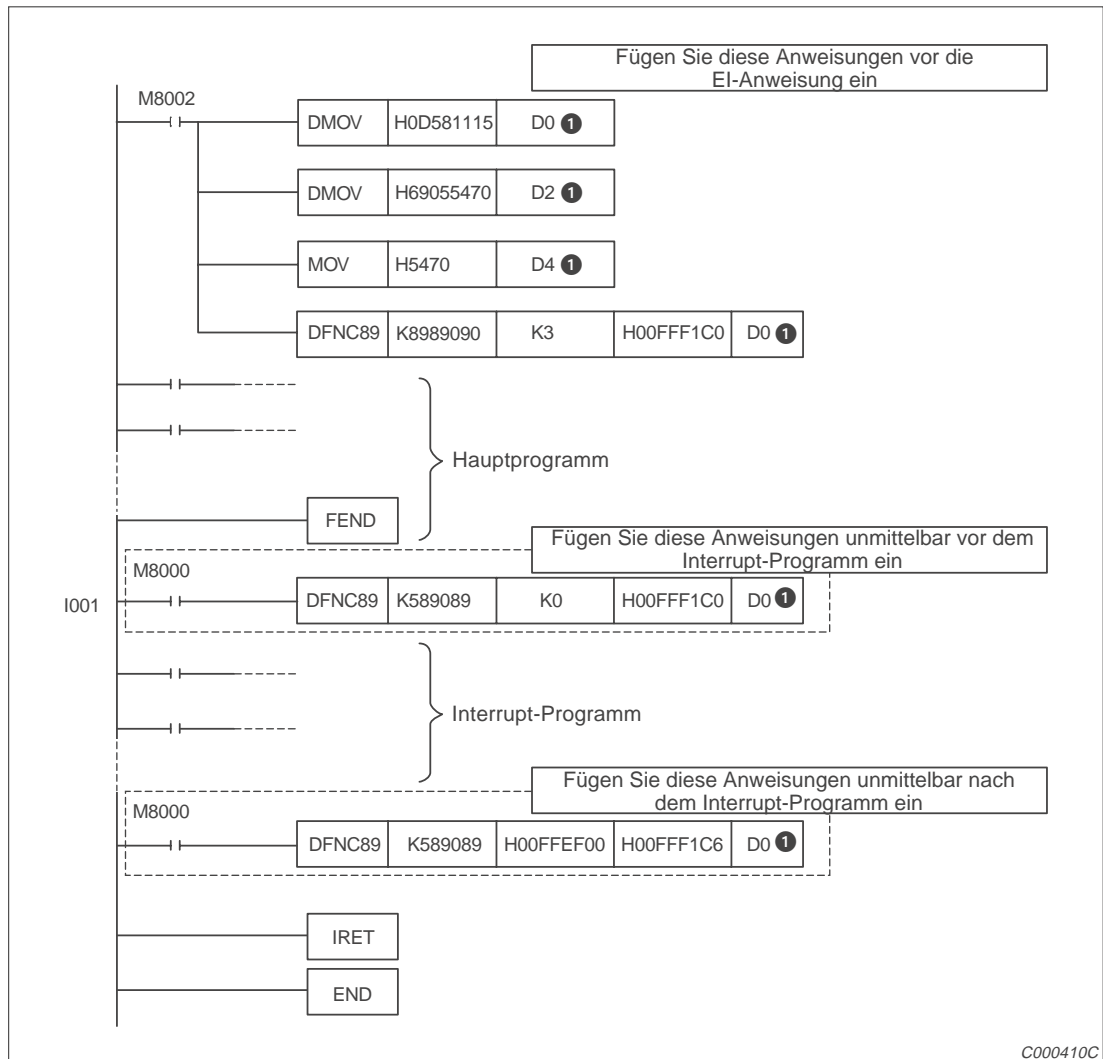


Abb. 3-29: Notwendige Programmiererweiterungen für FX1S (V1.00 bis V1.30) bei Verwendung von Interrupt-Pointern

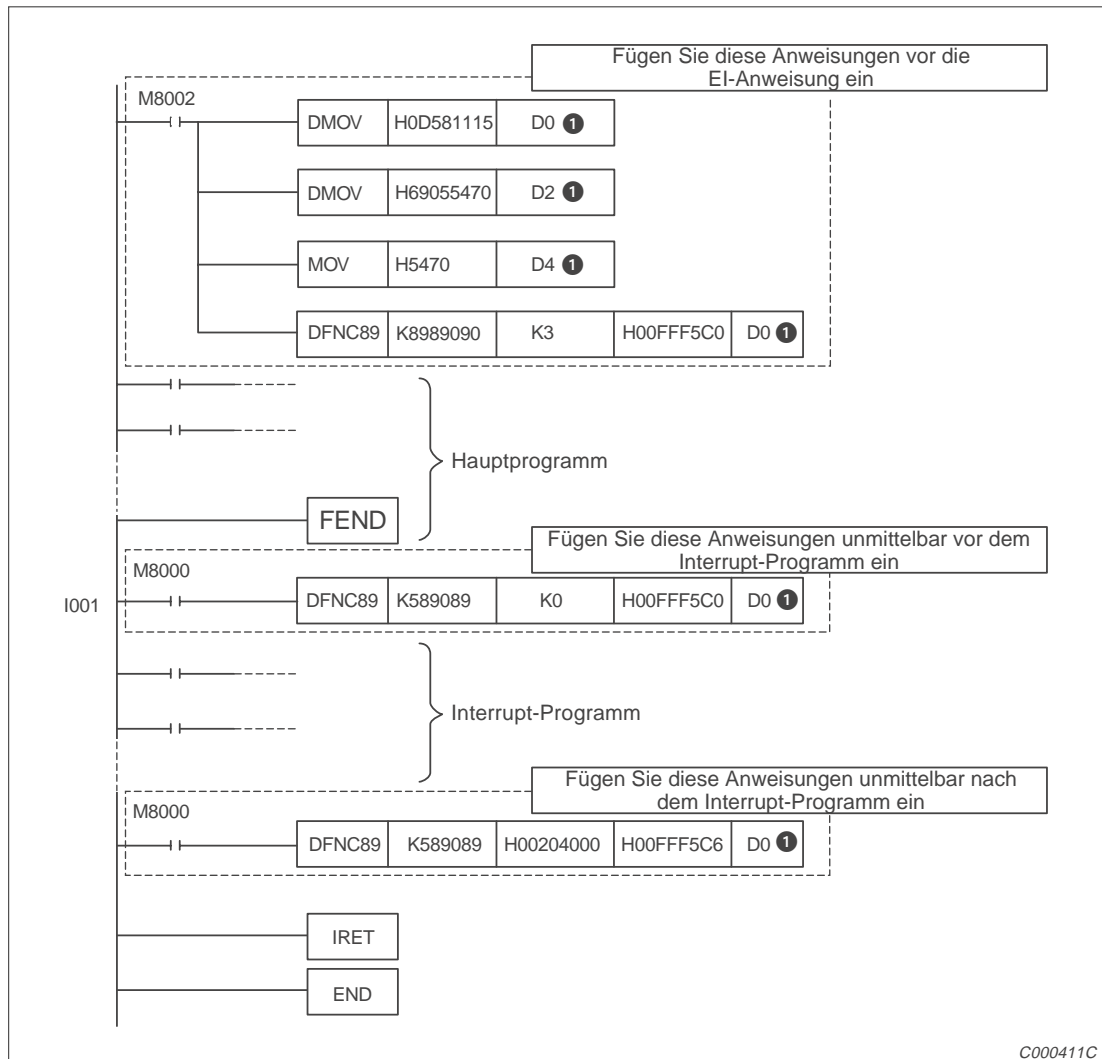
- ① Die Register D0 bis D4 sind durch diese Programmteile belegt und können nicht im SPS-Programm verwendet werden.

HINWEIS

Bei einer FX1S ab der Versionsnummer 1.40 sind diese Programmiererweiterungen nicht notwendig.

FX1N

Bei einer FX1N der Version 1.00 muss bei Verwendung von Interrupt-Pointern das Programm mit den nachfolgend dargestellten Anweisungen erweitert werden.



C000411C

Abb. 3-30: Notwendige Programmerweiterungen für FX1S (V1.00 bis V1.30) bei Verwendung von Interrupt-Pointern

① Die Register D0 bis D4 sind durch diese Programmteile belegt und können nicht im SPS-Programm verwendet werden.

HINWEIS

Bei einer FX1N ab der Versionsnummer 1.10 sind diese Programmerweiterungen nicht notwendig.

FX2N**MELSEC FX2N:**

Es stehen 9 Interrupt-Pointer zur Verfügung. Die Adressierung eines Interrupt-Pointers müssen Sie wie folgt vornehmen, wobei zwischen drei Gruppen unterschieden wird:

❶ Eingangs-Interrupt: I ❶ 0 ❷

❶ Adresse 0 bis 5

Jede Adresse darf nur einmal verwendet werden.

❷ 0: Interrupt bei abfallender Flanke

1: Interrupt bei ansteigender Flanke

Beispiel ▾

Interrupt-Pointer: I001

Das über den Interrupt-Pointer aufgerufene Interrupt-Programm wird bei ansteigender Flanke von X0 ausgeführt.

Die Rückkehr ins Hauptprogramm erfolgt, nachdem die IRET-Anweisung ausgeführt wurde.

△

❷ Timer-Interrupt: T ❶ ❷

❶ Adresse 6 bis 8

Jede Adresse darf nur einmal verwendet werden.

❷ 10 bis 99 ms

Beispiel ▾

Interrupt-Pointer: I610

Das über den Interrupt-Pointer I610 aufgerufene Interrupt-Programm wird in Intervallen von 10 ms ausgeführt.

Die Rückkehr ins Hauptprogramm erfolgt, nachdem die IRET-Anweisung ausgeführt wurde.

△

HINWEIS

Interrupt-Pointer werden nach einer FEND-Anweisung programmiert. Dabei dürfen nicht mehr als 9 Interrupt-Pointer eingesetzt werden. Mehr als zwei Verzweigungsebenen sind nicht zulässig.

❸ Counter-Interrupt: I 0 ❶ 0

❶ Adresse 1 bis 6

Counter-Interrupts können als Operanden zum Setzen (HSCS, FNC 53) oder Rücksetzen (HSCR, FNC 54) durch High-Speed-Counter verwendet werden. Zum Ausschalten des Counter-Interrupts schalten Sie den Sondermerker M8059 ein.

Beispiel ▾

Interrupt-Pointer: I030

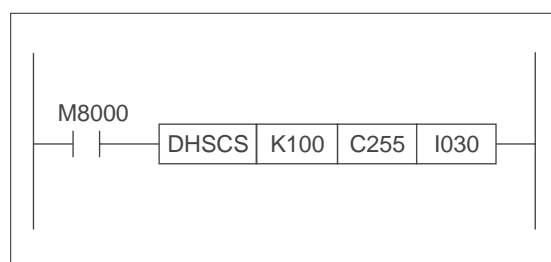


Abb. 3-31:

Programmierbeispiel zum Einsatz eines Counter-Interrupts

C000333C

Das über den Interrupt-Pointer I030 aufgerufene Interrupt-Programm wird ausgeführt, sobald der Wert des High-Speed-Counters C255 den in K100 angegebenen Wert erreicht.

△

HINWEIS

Bitte beachten Sie Absatz 6.7.4 mit den näheren Informationen zum Einsatz der Befehle zum Setzen oder Zurücksetzen durch High-Speed-Counter.

Ausschalten beliebiger Interrupts

Sie können beliebige Interrupts durch Einschalten der zugehörigen Sondermerker zeitweilig oder permanent Ausschalten. Die entsprechenden Sondermerker werden in Kapitel 6 angegeben. Für alle Steuerungen ist der erste Sondermerker M8050, der den Interrupt I0^{①②} ausschaltet.

HINWEISE

Setzen Sie nie einen Sondermerker, ohne sich seiner Funktion sicher zu sein. Nicht alle Steuerungen arbeiten immer mit den gleichen Sondermerkern.

High-Speed-Counter-Interrupts können immer nur als einzelne Gruppe mit dem Sondermerker M8059 ausgeschaltet werden.

3.11 Nesting

Mit Hilfe von Nesting-Operanden können Verzweigungsebenen innerhalb eines Programms realisiert werden. Nesting-Operanden werden in Zusammenhang mit den MC- und MCR-Anweisungen eingesetzt.

Der genaue Einsatz der Nesting-Operanden wird in Abs. 4.9 in der Beschreibung zu den Kontrollbedingungen (MC, MCR) beschrieben.

3.11.1 Nesting-Operanden adressieren

Es stehen acht Nesting-Operanden N0 bis N7 zur Verfügung.

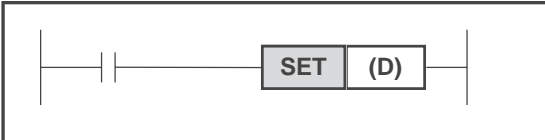
4 Grundbefehlssatz

4.1 Allgemeine Hinweise

Dieses Kapitel beschreibt den Grundbefehlssatz der FX-Familie. Mit den Anweisungen aus dem Grundbefehlssatz lassen sich alle logischen Grundverknüpfungen programmieren. Die Anweisungen aus dem Grundbefehlssatz können jeweils nur eine Operandenadresse ansprechen.

4.1.1 Erläuterung der Grundbefehlssatztabellen

Alle Grundbefehle sind in einer tabellarischen Form auf den nächsten beiden Seiten aufgelistet. Dieser Abschnitt erläutert kurz die Struktur der Übersichtstabellen.


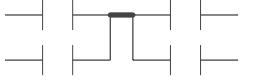

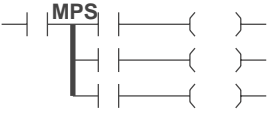
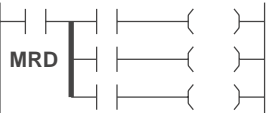
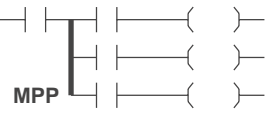
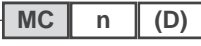



②	①																
	<table border="1"> <thead> <tr> <th colspan="4" style="background-color: black; color: white;">SET</th> </tr> <tr> <th colspan="4">③ Setzen; Operanden setzen</th> </tr> <tr> <th style="text-align: center;">CPU</th> <th style="text-align: center;">FX1S</th> <th style="text-align: center;">FX1N</th> <th style="text-align: center;">FX2N</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">④</td> <td style="text-align: center;">●</td> <td style="text-align: center;">●</td> <td style="text-align: center;">●</td> </tr> </tbody> </table>	SET				③ Setzen; Operanden setzen				CPU	FX1S	FX1N	FX2N	④	●	●	●
SET																	
③ Setzen; Operanden setzen																	
CPU	FX1S	FX1N	FX2N														
④	●	●	●														
<table border="1"> <thead> <tr> <th style="text-align: left;">Operanden</th> <th style="text-align: left;">Programmschritte</th> <th style="text-align: left;">Bemerkung</th> </tr> </thead> <tbody> <tr> <td rowspan="2">Y, M, S</td> <td>SET (Operanden Y, M)</td> <td style="text-align: center;">1</td> </tr> <tr> <td>SET (Sondermerker S)</td> <td style="text-align: center;">2</td> </tr> </tbody> </table>		Operanden	Programmschritte	Bemerkung	Y, M, S	SET (Operanden Y, M)	1	SET (Sondermerker S)	2								
Operanden	Programmschritte	Bemerkung															
Y, M, S	SET (Operanden Y, M)	1															
	SET (Sondermerker S)	2															
⑤	⑥																

- ① **Anweisung**
An dieser Stelle wird der Name der Anweisung angegeben, der bei der Programmierung in Anweisungsliste eingesetzt wird.
- ② **Kontaktplansymbol**
Das Kontaktplansymbol wird bei der Kontaktplanprogrammierung verwendet. Das Kontaktplansymbol besteht aus der Anweisung und den einsetzbaren Operanden.
- ③ **Bedeutung**
Hier finden Sie eine kurze Beschreibung zur Bedeutung der Anweisung.
- ④ **CPU**
Hier wird die MELSEC SPS-Serie durch einen ● gekennzeichnet, mit der diese Anweisung ausführbar ist.
- ⑤ **Operanden**
In diesem Feld werden die in Zusammenhang mit der Anweisung einsetzbaren Operanden angegeben.
- ⑥ **Programmschritte**
Es wird die Anzahl von Programmschritten angegeben, die bis zur vollständigen Ausführung der Anweisung erforderlich sind.



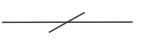


Übersicht der Grundbefehle

Anweisung	Kontaktplan-symbol	Bedeutung	Operanden	Programmschritte	Referenz
LD		LADE; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.2
LDI		LADE NICHT; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.2
OUT		AUSGABE; Ausgabe, Zuweisung eines Verknüpfungsergebnisses	Y, M, S, T, C	Y, M: 1 S, Sondermerker: 2 T: 3, C: 3 C(32 Bit): 5	Abs. 4.3
AND		UND; UND-Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.4
ANI		UND Nicht; UND-Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.4
OR		ODER; ODER-Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.5
ORI		ODER Nicht; ODER-Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.5
LDP		LADE; (gepulst) Beginn einer Verknüpfung mit Abfrage der ansteigenden Flanke;	X, Y, M, S, T, G	2	Abs. 4.6
LDF		LADE; (gepulst) Beginn einer Verknüpfung mit Abfrage der abfallenden Flanke	X, Y, M, S, T, G	2	Abs. 4.6
ANP		UND; (gepulst) UND-Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, G	2	Abs. 4.7
ANF		UND; (gepulst) UND-Verknüpfung mit Abfrage der abfallenden Flanke	X, Y, M, S, T, G	2	Abs. 4.7
ORP		ODER; ODER-Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, G	2	Abs. 4.8

Tab. 4-1: Grundbefehlsübersicht (Teil 1)

Anweisung	Kontaktplan-symbol	Bedeutung	Operanden	Programmschritte	Referenz
ORF		ODER; ODER-Verknüpfungen mit Abfrage der abfallende Flanke	X, Y, M, S, T, G	2	Abs. 4.8
ANB		UND-Block; Koppelbefehl: Reihenschaltung von Parallelverknüpfungen	—	1	Abs. 4.9
ORB		ODER-Block; Koppelbefehl: Parallelschaltung von Reihenverknüpfungen	—	1	Abs. 4.10
MPS		Push Down Stack; Abspeichern eines Verknüpfungsergebnisses	—	1	Abs. 4.11
MRD		Read Down Stack; Lesen eines Verknüpfungsergebnisses	—	1	Abs. 4.11
MPP		Pop Up Stack; Lesen und Löschen des Verknüpfungsspeichers	—	1	Abs. 4.11
MC		Master Control; Setzen einer Kontrollbedingung	Y, M, keine Sondermerker	3	Abs. 4.12
MCR		Master Control Reset; Rücksetzen einer Kontrollbedingung	N	2	Abs. 4.12
SET		Setzen; Operanden setzen	Y, M, S	Y, M: 1 S, Sondermerker: 2	Abs. 4.13
RST		Rücksetzen; Operanden rücksetzen	Y, M, S, D V, Z, T, C	Y, M: 1 D, V, Z, Sondermerker: 3 T, C: 2	Abs. 4.13

Tab. 4-2: Grundbefehlsübersicht (Teil 2)

Anweisung	Kontaktplan-symbol	Bedeutung	Operanden	Programmschritte	Referenz
PLS		Impulserzeugung; Erzeugen eines einmaligen Impulses bei ansteigender Flanke	Y, M	2	Abs. 4.14
PLF		Impulserzeugung; Erzeugen eines einmaligen Impulses bei abfallender Flanke	Y, M	2	Abs. 4.14
INV		Inversion; Umkehrung von Verarbeitungsergebnissen	—	1	Abs. 4.15
NOP		Leerzeile; Leerzeile ohne Funktion	—	1	Abs. 4.16
END		Ende; SPS-Programmende	—	1	Abs. 4.17

Tab. 4-3: Grundbefehlsübersicht (Teil 3)

4.2 Beginn von Verknüpfungen (LD, LDI)

	LD		
	LADE; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „1“		
	CPU	FX1S ●	FX1N ●
	LDI		
	LADE NICHT; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „0“		
	CPU	FX1S ●	FX1N ●
Operanden	Programmschritte		Bemerkung
X, Y, M, S, T, C	LD	1	
	LDI	1	

Funktion

Programmieren eines Verknüpfungsbegins

Beschreibung

- Der Beginn einer Verknüpfung wird mit einer LD- oder LDI-Anweisung programmiert.
- Die Programmierung eines Strompfades beginnt immer mit einer LD- oder LDI-Anweisung.
- Die LD- und LDI-Anweisung wird auch im Zusammenhang mit der ANB- und ORB-Anweisung zum Starten einer Verzweigung benutzt (siehe auch Abs. 4.6 und 4.7).

Beispiel ▾ Einsatz der Anweisungen LD, LDI

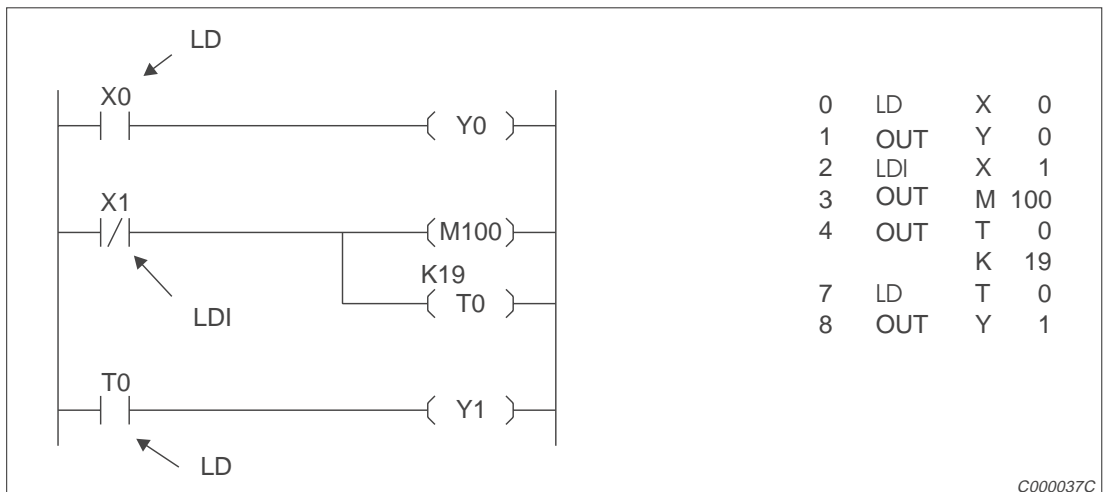


Abb. 4-1: Programmierbeispiel zum Einsatz der Anweisungen LD, LDI

Der Eingang X0 wird auf den Signalzustand „1“ abgefragt. Der Ausgang Y0 wird auf den Signalzustand „1“ geschaltet, sobald der Eingang X0 ein „1“-Signal erhält.

Der Eingang X1 wird auf den Signalzustand „0“ abgefragt. Der Merker M100 erhält den Signalzustand „1“, und die eingestellte Zeit des Timers T0 beginnt abzulaufen, sobald der Eingang X1 ein „0“-Signal erhält.

Nach Ablauf des eingestellten Zeitsollwertes (19 x 100 ms = 1,9 s) schaltet der Timer T0 den Ausgang Y1 auf den Signalzustand „1“. △

4.3 Ausgabe eines Verknüpfungsergebnisses (OUT)

		OUT				
		AUSGABE; Ausgabe, Zuweisung eines Verknüpfungsergebnisses				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden		Programmschritte			Bemerkung	
Y, M, S, T, C		Y, M	1	T, C (16 Bit)	3	
		S,	2	C (32 Bit)	5	

Funktion

Zuweisung eines Signalzustandes in Abhängigkeit vom Verknüpfungsergebnis

Beschreibung

- Mit der OUT-Anweisung kann die Programmierung eines Strompfades abgeschlossen werden.
- Die Programmierung mehrerer OUT-Anweisungen als Ergebnis einer Verknüpfung ist möglich.
- Das durch die OUT-Anweisung dargestellte Verknüpfungsergebnis kann in den nachfolgenden Programmschritten als Eingangssignalzustand eingesetzt werden.
- Das Verknüpfungsergebnis, das durch die OUT-Anweisung dargestellt wird, ist nur solange aktiv, wie die Einschaltbedingung gegeben ist.

Beispiel ▾

Einsatz der Anweisung OUT

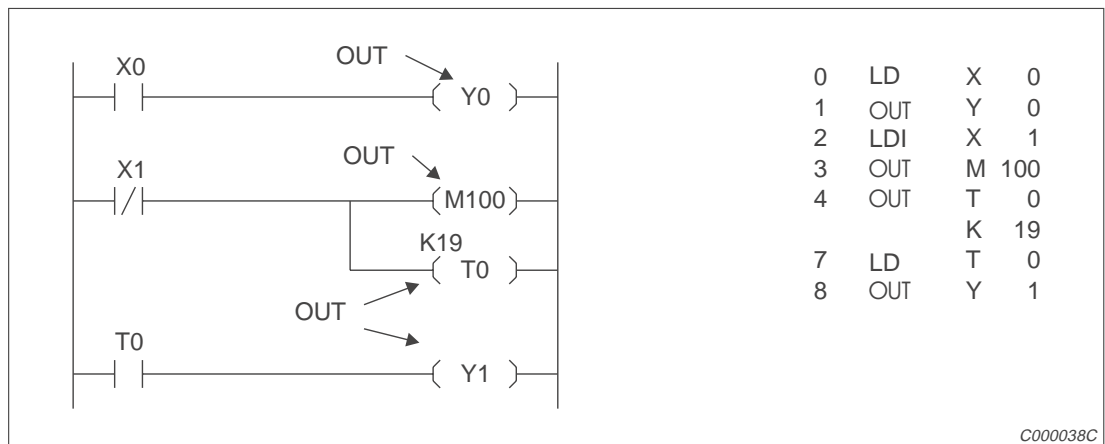


Abb. 4-2: Programmierbeispiel zum Einsatz der Anweisung OUT

Der Eingang X0 wird auf den Signalzustand „1“ abgefragt. Der Ausgang Y0 wird auf den Signalzustand „1“ geschaltet, sobald der Eingang X0 ein „1“-Signal erhält.

Der Eingang X1 wird auf den Signalzustand „0“ abgefragt. Der Merker M100 und der Timer T0 werden auf den Signalzustand „1“ geschaltet, sobald der Eingang X1 ein „0“-Signal erhält.

Nach Ablauf des eingestellten Zeitsollwertes (19 x 100 ms = 1,9 s) schaltet der Timer T0 den Ausgang Y1 auf den Signalzustand „1“.

Anzahl der Programmschritte beim Einsatz von Timern und Countern

Bei OUT-Anweisungen, die in Zusammenhang mit Timern und Countern programmiert werden, ist darauf zu achten, daß es sich um Zweischrittanweisungen handelt. Im zweiten Programmierschritt erfolgt die Einstellung des Zeit- bzw. des Zählerwertes. Dies geschieht durch die Eingabe einer dezimalen Konstante K.

Die Programmierung von Timern und Countern wird in Abs. 3.4. und Abs. 3.5 ausführlich beschrieben.

Doppelbelegung von Ausgängen

Bei der Programmierung von doppelten Ausgangsbelegungen kann es zu Problemen beim Programmablauf kommen. Das folgende Beispiel verdeutlicht diese Problematik.

Beispiel ▾

Doppelbelegung eines Ausgangs

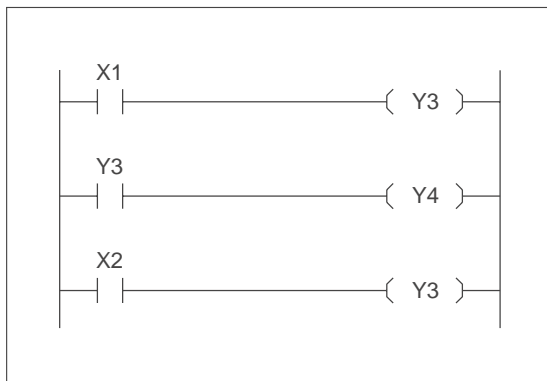


Abb. 4-3:

Programmierbeispiel zur Doppelbelegung eines Ausgangs

C000050C

Es wird davon ausgegangen, daß der Eingang X1 eingeschaltet (Signal „1“) und der Eingang X2 ausgeschaltet (Signal „0“) ist.

Der erste Ausgang Y3 wird durch den eingeschalteten Eingang X1 aktiviert. Im Prozeßabbild der Ausgänge ist Y3 eingeschaltet. Entsprechend wird auch der Ausgang Y4 aktiviert.

Im nächsten Schritt wird der Ausgang Y3 wieder deaktiviert, da der Eingang X2 ausgeschaltet ist. Im Prozeßabbild der Ausgänge ist Y3 ausgeschaltet.

Diese Programmsequenz hat zur Folge, daß Y3 ausgeschaltet und Y4 eingeschaltet ist.

In Abs. 2.1 wird die Abarbeitung eines SPS-Programms detailliert beschrieben. △

HINWEIS

Vermeiden Sie eine Doppelbelegung von Ausgängen, da dies zu Programmablaufstörungen führen kann.

Beispiel ▾

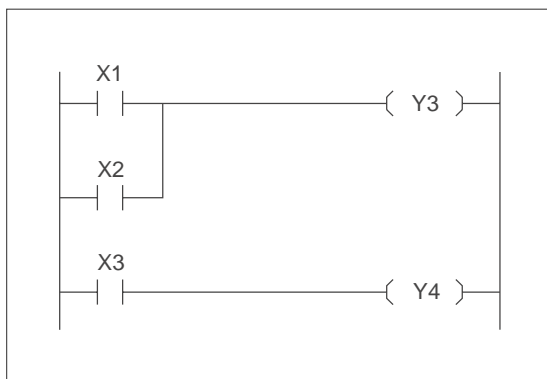


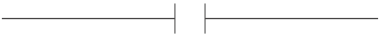

Abb. 4-4:

Programmierbeispiel

C000051C

△

4.4 UND-Verknüpfungen (AND, ANI)

	AND		
	UND; UND-Verknüpfung mit Abfrage auf Signalzustand „1“		
	CPU	FX1S	FX1N
	●	●	●
	ANI		
	UND NICHT; UND-Verknüpfung mit Abfrage auf Signalzustand „0“		
	CPU	FX1S	FX1N
	●	●	●
Operanden	Programmschritte		Bemerkung
X, Y, M, S, T, C	AND-Anweisung	1	
	ANI-Anweisung	1	

Funktion

Programmieren von logischen UND-Verknüpfungen

Beschreibung

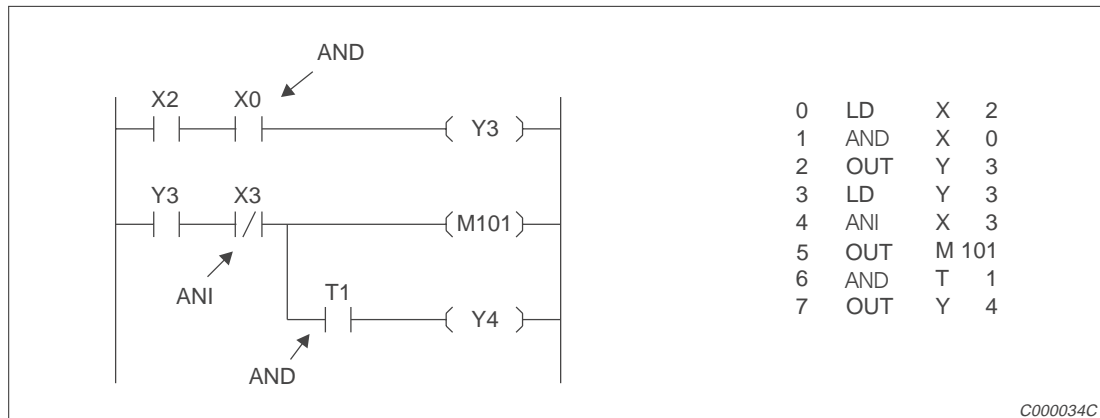
- Die Anweisungen AND und ANI werden für die Reihenschaltung von Kontakten (logische UND-Verknüpfung) eingesetzt.
- Beide Anweisungen stellen logische Verknüpfungen dar und dürfen daher nicht am Anfang eines Strompfades programmiert werden. Der Beginn einer Verknüpfung wird mit einer LD- oder LDI-Anweisung programmiert (siehe Abs. 4.2).
- Wenn Sie mehrere aufeinanderfolgende Blockschaltungen in Reihe schalten möchten, können Sie auch die ANB-Anweisung verwenden (siehe Abs. 4.6).

HINWEIS

Es dürfen nur 10 Kontakte pro Strompfad und maximal 24 Strompfade pro Spule programmiert werden.

Beispiel ▾

Einsatz der Anweisungen AND, ANI

**Abb. 4-5:** Programmierbeispiel zum Einsatz der Anweisungen AND und ANI

Der Ausgang Y3 weist den Signalzustand „1“ auf, wenn folgende Bedingungen erfüllt sind:

- der Eingang X2 weist den Signalzustand „1“ auf
UND
- der Eingang X0 weist den Signalzustand „1“ auf.

Der Merker M101 weist den Signalzustand „1“ auf, wenn folgende Bedingungen erfüllt sind:



- der Ausgang Y3 weist den Signalzustand „1“ auf,
UND
- der Eingang X3 weist den Signalzustand „0“ auf.

Der Ausgang Y4 weist den Signalzustand „1“ auf, wenn folgende Bedingungen erfüllt sind:

- der Ausgang Y3 weist den Signalzustand „1“ auf,
UND
- der Eingang X3 weist den Signalzustand „0“ auf,
UND
- der Timerkontakt T1 weist den Signalzustand „1“ auf.

△

4.5 ODER-Verknüpfungen (OR, ORI)

	OR		
	ODER; ODER-Verknüpfung mit Abfrage auf Signalzustand „1“		
	CPU	FX1S	FX1N
	●	●	●
	ORI		
	ODER NICHT; ODER-Verknüpfung mit Abfrage auf Signalzustand „0“		
	CPU	FX1S	FX1N
	●	●	●
Operanden	Programmschritte		Bemerkung
X, Y, M, S, T, C	OR-Anweisung	1	
	ORI-Anweisung	1	

Funktion

Programmieren von logischen ODER-Verknüpfungen

Beschreibung

- Die Anweisungen OR und ORI werden für die Parallelschaltung von Kontakten (logische ODER-Verknüpfungen) eingesetzt.
- Beide Anweisungen stellen logische Verknüpfungen dar und dürfen daher nicht am Anfang eines Strompfades programmiert werden. Der Beginn einer Verknüpfung wird mit einer LD- oder LDI-Anweisung programmiert (siehe Abs. 4.2)
- Wenn Sie mehrere aufeinanderfolgende Blockschaltungen parallel schalten möchten, können Sie auch die ORB-Anweisung verwenden (siehe Abs. 4.7)

HINWEIS | Sie sollten nicht mehr als 24 Strompfade als Parallelschaltung programmieren.

Beispiel ▾

Einsatz der Anweisungen OR, ORI

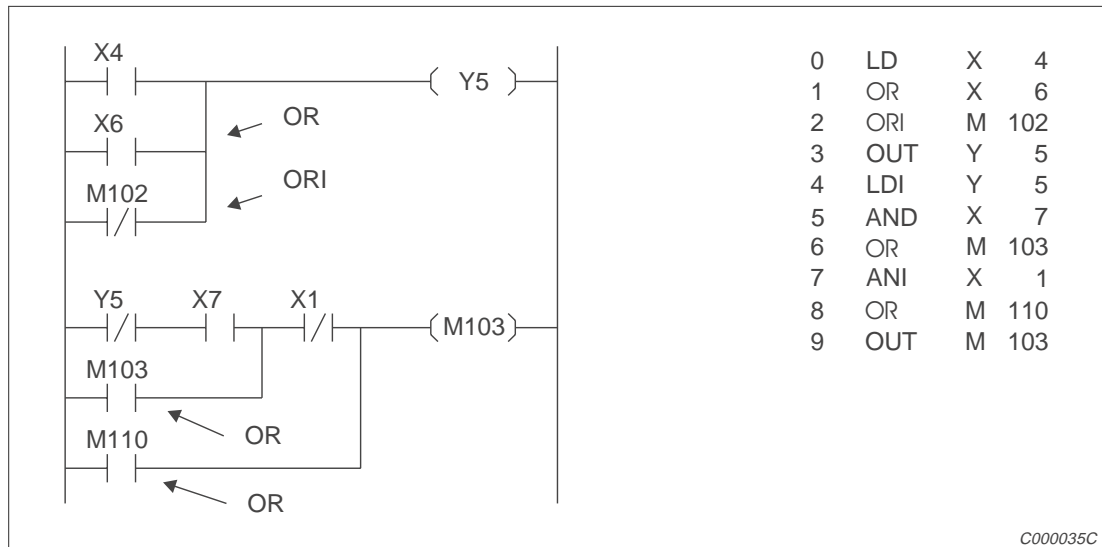


Abb. 4-6: Programmierbeispiel zum Einsatz der Anweisungen OR und ORI

Der Ausgang Y5 weist den Signalzustand „1“ auf, wenn eine der folgenden Bedingungen erfüllt ist:

- Der Eingang X4 weist den Signalzustand „1“ auf,
ODER
- der Eingang X6 weist den Signalzustand „1“ auf,
ODER
- der Merker M102 weist den Signalzustand „0“ auf.

Der Merker M103 weist den Signalzustand „1“ auf, wenn eine der folgenden Bedingungen erfüllt ist:



- Der Ausgang Y5 weist den Signalzustand „0“ auf,
UND
der Eingang X7 weist den Signalzustand „1“ auf,
UND
der Eingang X1 weist den Signalzustand „0“ auf,

ODER
- der Eingang X1 weist den Signalzustand „0“ auf,
UND
der Merkerkontakt M103 weist den Signalzustand „1“ auf (Merker wird über eine Selbsthaltung auf „1“-Signal gesetzt),

ODER
- der Merkerkontakt M110 weist den Signalzustand „1“ auf.



4.6 Gepulster Beginn von Verknüpfungen (LDP, LDF)

	LDP			
	LADE (gepulst); Beginn einer Verknüpfung mit Abfrage der ansteigenden Flanke			
	CPU	FX1S	FX1N	FX2N
	●	●	●	
	LDF			
	LADE (gepulst); Beginn einer Verknüpfung mit Abfrage der abfallenden Flanke			
	CPU	FX1S	FX1N	FX2N
	●	●	●	
Operanden	Programmschritte		Bemerkung	
X, Y, M, S, T, C	LDP	2		
	LDF	2		

Funktion

Programmieren eines gepulsten Verknüpfungsbeginns

Beschreibung

- Der Beginn einer gepulsten Verknüpfung wird mit einer LDP- (ansteigende Flanke) oder einer LDF-Anweisung (abfallende Flanke) programmiert.
- Die LDP- und LDF-Anweisungen müssen am Beginn eines Strompfades programmiert werden.
- Die LDP- und LDF-Anweisungen werden auch im Zusammenhang mit der ANB- und ORB-Anweisung zum Starten einer Verzweigung benutzt (siehe Abs. 4.9 und 4.10)
- Die LDP-Anweisung bleibt nach der positiven Flanke für einen Programmzyklus gesetzt.
- Die LDF-Anweisung bleibt nach der negativen Flanke für einen Programmzyklus gesetzt.

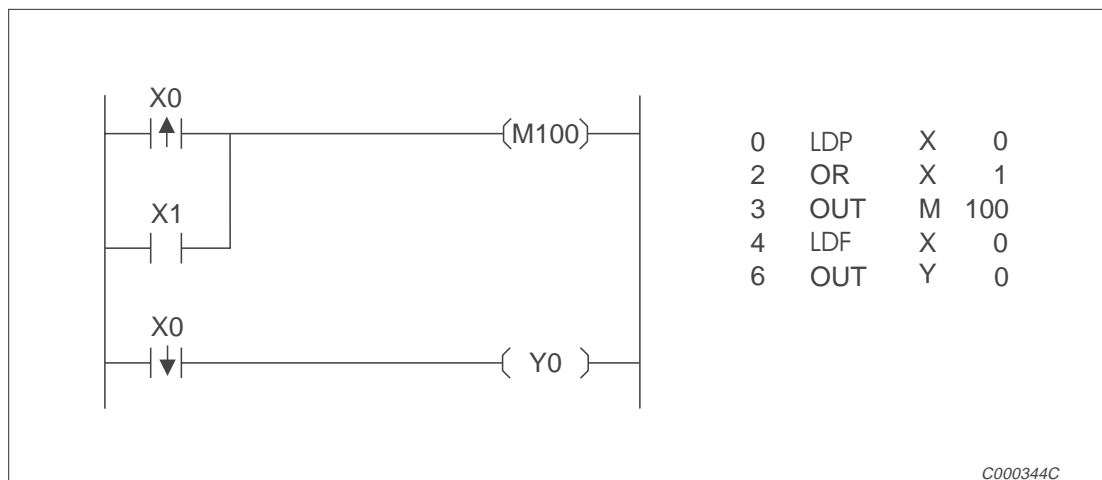
HINWEISE

Werden im Zusammenhang mit der LDP- oder LDF-Anweisung die gepulsten Merker M2800 bis M3071 verwendet, wird bei mehrmaliger Programmierung des selben gepulsten Merkers in einem Programm nur der erste Merker verarbeitet. Diese Eigenschaft wird im Zusammenhang mit der STL-Programmierung eingesetzt (siehe Abs. 5).

Die Funktionen der LD-, AND-, OR- etc. Anweisungen bleiben unverändert.

Beispiel ▾

Einsatz der Anweisungen LDP, LDF

**Abb. 4-7:** Programmierbeispiel zum Einsatz der Anweisung LDP, LDF

Der Merker M100 wird für die Einschaltdauer von X1 oder bei positiver Flanke von X0 gesetzt.

Der Ausgang Y0 wird bei negativer Flanke von X0 gesetzt.

△

4.7 Gepulste UND-Verknüpfungen (ANP, ANF)

	ANP		
	UND-Verknüpfung (gepulst), UND-Verknüpfung mit Abfrage der ansteigenden Flanke		
	CPU	FX1S	FX1N
	●	●	●
	ANF		
	UND-Verknüpfung (gepulst), UND-Verknüpfung mit Abfrage der abfallenden Flanke		
	CPU	FX1S	FX1N
	●	●	●
Operanden	Programmschritte		Bemerkung
X, Y, M, S, T, C	ANP	2	
	ANF	2	

Funktion

Programmieren einer gepulsten UND-Verknüpfung

Beschreibung

- Eine gepulste UND-Verknüpfung wird mit einer ANP- (ansteigende Flanke) oder einer ANF-Anweisung (abfallende Flanke) programmiert.
- Die ANP- und ANF-Anweisungen können wie die AND- und ANI-Anweisungen verwendet werden.
- Die ANP-Anweisung mit der positiven Flanke verarbeitet.
- Die ANF-Anweisung wird mit der negativen Flanke verarbeitet.

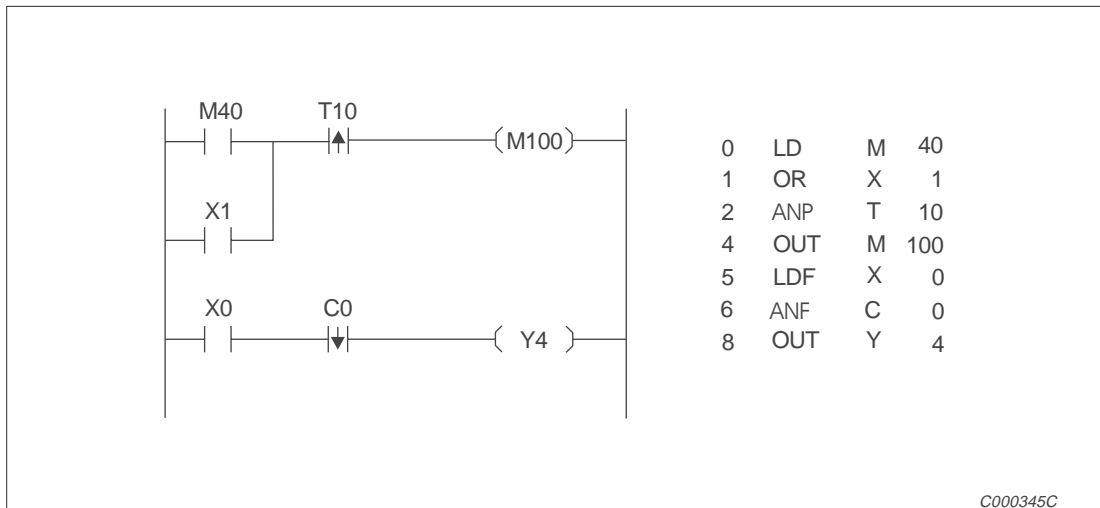
HINWEISE

Werden im Zusammenhang mit der ANP- oder ANF-Anweisung die gepulsten Merker M2800 bis M3071 verwendet, wird bei mehrmaliger Programmierung des selben gepulsten Merkers in einem Programm nur der erste Merker verarbeitet. Diese Eigenschaft wird im Zusammenhang mit der STL-Programmierung eingesetzt (siehe Abs. 5).

Die Funktionen der LD-, AND-, OR- etc. Anweisungen bleiben unverändert.

Beispiel ▾

Einsatz der Anweisungen ANP, ANF



**Abb. 4-8:** Programmierbeispiel zum Einsatz der Anweisungen ANP, ANF

Der Merker M100 wird bei gesetztem Merker M40 oder gesetztem Eingang X1 und der ansteigenden Flanke vom Timer-Kontakt T10 gesetzt.

Der Ausgang Y4 wird bei gesetztem Eingang X0 und negativer Flanke des Counter-Kontaktes C0 gesetzt.

△

4.8 Gepulste ODER-Verknüpfungen (ORP, ORF)

	ORP		
	ODER-Verknüpfung (gepulst); ODER-Verknüpfung mit Abfrage der ansteigenden Flanke		
	ORF		
	ODER-Verknüpfung (gepulst); ODER-Verknüpfung mit Abfrage der abfallenden Flanke		
CPU	FX1S	FX1N	FX2N
	●	●	●
CPU	FX1S	FX1N	FX2N
	●	●	●
Operanden	Programmschritte		Bemerkung
X, Y, M, S, T, C	ORP	2	
	ORF	2	

Funktion

Programmieren einer gepulsten ODER-Verknüpfung

Beschreibung

- Eine gepulste ODER-Verknüpfung wird mit einer ORP- (ansteigende Flanke) oder einer ORF-Anweisung (abfallende Flanke) programmiert.
- Die ORP- und ORF-Anweisungen können wie die OR- und ORI-Anweisungen verwendet werden.
- Die ORP-Anweisung wird mit der positiven Flanke verarbeitet.
- Die ORF-Anweisung wird mit der negativen Flanke verarbeitet.

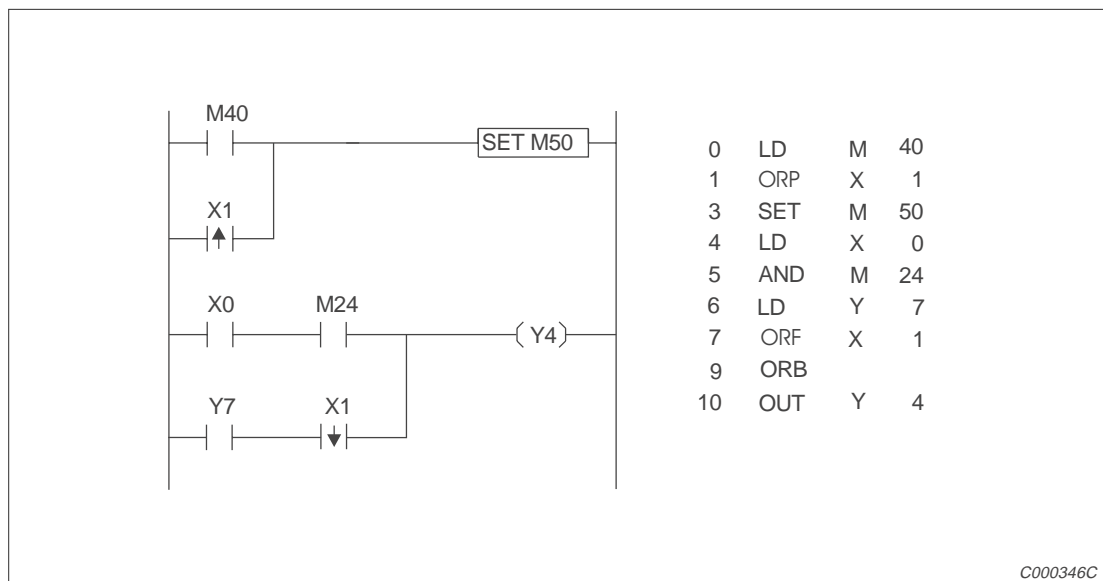
HINWEISE

Werden im Zusammenhang mit der ORP- oder ORF-Anweisung die gepulsten Merker M2800 bis M3071 verwendet, wird bei mehrmaliger Programmierung des selben gepulsten Merkers in einem Programm nur der erste Merker verarbeitet. Diese Eigenschaft wird im Zusammenhang mit der STL-Programmierung eingesetzt (siehe Abs. 5).

Die Funktionen der LD-, AND-, OR- etc. Anweisungen bleiben unverändert.

Beispiel ▾

Einsatz der Anweisungen ORP, ORF

**Abb. 4-9:** Programmierbeispiel zum Einsatz der Anweisungen ORP, ORF

Der Merker M50 wird über die SET-Anweisung bei gesetztem Merker M40 oder der ansteigenden Flanke des Eingangs X1 gesetzt.

Der Ausgang Y4 wird bei gesetztem Eingang X0 und gesetztem Merker M24 oder bei gesetztem Ausgang Y7 und abfallender Flanke von X1 gesetzt.

△

4.9 UND-Block-Verknüpfung (ANB)

	ANB			
	UND-BLOCK; Koppelbefehl: Reihenschaltung von Parallelverknüpfungen			
	CPU	FX1S	FX1N	FX2N
		●	●	●
Operanden	Programmschritte			Bemerkung
—	ANB-Anweisung	1		

Funktion

Reihenschaltung von parallelen Block-Verknüpfungen

Beschreibung

- Einzelne parallelgeschaltete Blöcke werden separat eingegeben. Um diese Blöcke anschließend in Reihe zu schalten, ist nach jedem Block die ANB-Anweisung zu programmieren.
- Der Beginn einer Verzweigung wird mit der Anweisung LD bzw. LDI programmiert (siehe Abs. 4.2).
- Die ANB-Anweisung ist eine unabhängige Anweisung und erfordert keine Angabe eines Operanden.
- Die ANB-Anweisung kann innerhalb des gesamten Programms beliebig oft programmiert werden.
- Im Kontaktplan wird die ANB-Anweisung als eine serielle Verbindung dargestellt. Die ANB-Anweisung erscheint automatisch in der Anweisungsliste, nachdem das Programm im Kontaktplan konvertiert wurde.

HINWEIS

Wenn Sie mehrere einzelne Blöcke direkt hintereinander programmieren, müssen Sie die Anzahl der LD- und LDI-Anweisungen und somit auch die Anzahl der ANB-Anweisungen auf 8 beschränken.

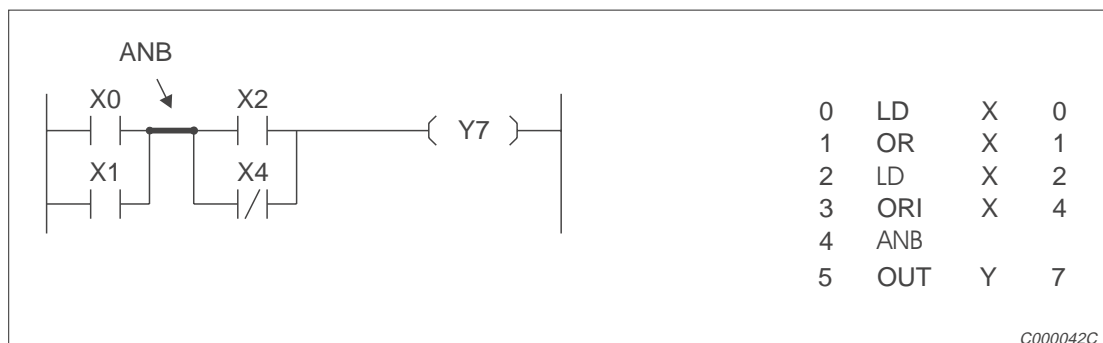


Abb. 4-10: Programmierbeispiel zum Einsatz der Anweisung ANB

4.10 ODER-Block-Verknüpfung (ORB)

	ORB			
	ODER-BLOCK ; Koppelbefehl: Parallelschaltung von Reihenverknüpfungen			
	CPU	FX1S	FX1N	FX2N
	●	●	●	
Operanden	Programmschritte			Bemerkung
—	ORB-Anweisung	1		

Funktion

Parallelschaltung von in Reihe geschalteten Block-Verknüpfungen

Beschreibung

- Werden mehrere in Reihe geschaltete Blöcke parallelgeschaltet, muß nach der Programmierung jedes einzelnen Blocks eine ORB-Anweisung eingegeben werden.
- Der Beginn einer Verzweigung wird mit der Anweisung LD bzw. LDI programmiert (siehe Abs. 4.2).
- Die ORB-Anweisung ist eine unabhängige Anweisung und erfordert keine Angabe eines Operanden.
- Die ORB-Anweisung kann innerhalb des gesamten Programms beliebig oft programmiert werden.
- Im Kontaktplan wird die ORB-Anweisung als eine parallele Verbindung dargestellt. Die ORB-Anweisung erscheint automatisch in der Anweisungsliste, nachdem das Programm im Kontaktplan konvertiert wurde.

HINWEIS

Wenn Sie mehrere einzelne Blöcke direkt hintereinander programmieren, müssen Sie die Anzahl der LD- und LDI-Anweisungen und somit auch die Anzahl der ORB-Anweisungen auf 8 beschränken.

	Empfohlene Programmierung	Ungünstige Programmierung
	0 LD X 0	0 LD X 0
	1 AND X 1	1 AND X 1
	2 LD X 2	2 LD X 2
	3 AND X 3	3 AND X 3
	4 ORB	4 LDI X 4
	5 LDI X 4	5 AND X 5
	6 AND X 5	6 ORB
	7 ORB	7 ORB
8 OUT Y 6	8 OUT Y 6	

C000043C

Abb. 4-11: Programmierbeispiel zum Einsatz der Anweisung ORB

4.11 Verarbeiten eines Verknüpfungsergebnisses (MPS, MRD, MPP)

	MPS		
	Push Down Stack; Abspeichern eines Verknüpfungsergebnisses		
	MRD		
	Read Down Stack; Lesen eines Verknüpfungsergebnisses		
	MPP		
	Pop Up Stack; Lesen und Löschen eines Verknüpfungsspeichers		
CPU	FX1S	FX1N	FX2N
	●	●	●
CPU	FX1S	FX1N	FX2N
	●	●	●
CPU	FX1S	FX1N	FX2N
	●	●	●
Operanden	Programmschritte		Bemerkung
—	MPS-Anweisung	1	
	MRD-Anweisung	1	
	MPP-Anweisung	1	

Funktion

Die Anweisungen MPS, MRD und MPP dienen dazu, Verknüpfungsebenen aufzubauen. Mit Hilfe dieser Anweisungen wird der Programmieraufwand erheblich reduziert.

Beschreibung

- Mit der Anweisung MPS wird das vorangegangene Verknüpfungsergebnis abgespeichert.
- Mit Hilfe der MRD-Anweisung werden mehrere Teilverzweigungen zwischen dem Beginn (MPS) und dem Ende (MPP) der Verzweigung möglich.
- Die letzte Teilverzweigung wird mit der MPP-Anweisung gestartet.
- Die mit einer MPS-Anweisung eröffnete Verzweigung muß immer mit einer MPP-Anweisung abgeschlossen werden.
- Alle drei Anweisungen erfordern keine Angabe eines Operanden.
- Im Kontaktplan werden diese Anweisungen nicht dargestellt. Erfolgt die Programmierung im Kontaktplan, werden die Verzweigungen wie bisher gesetzt. Die MPS-, MRD- und MPP-Anweisungen erscheinen automatisch in der Anweisungsliste, nachdem das Programm im Kontaktplan konvertiert wurde.

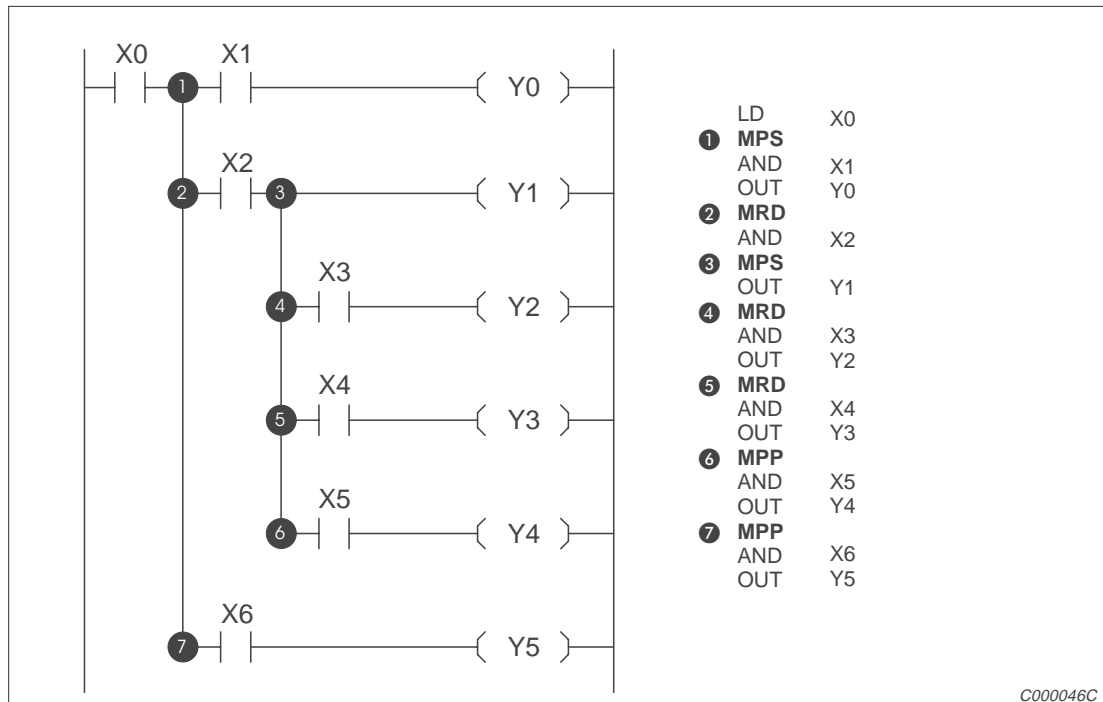
HINWEIS

| Es sind maximal 11 Verknüpfungsebenen zulässig.

Eine detaillierte Beschreibung der drei Anweisungen soll anhand der folgenden Programmierbeispiele gegeben werden.

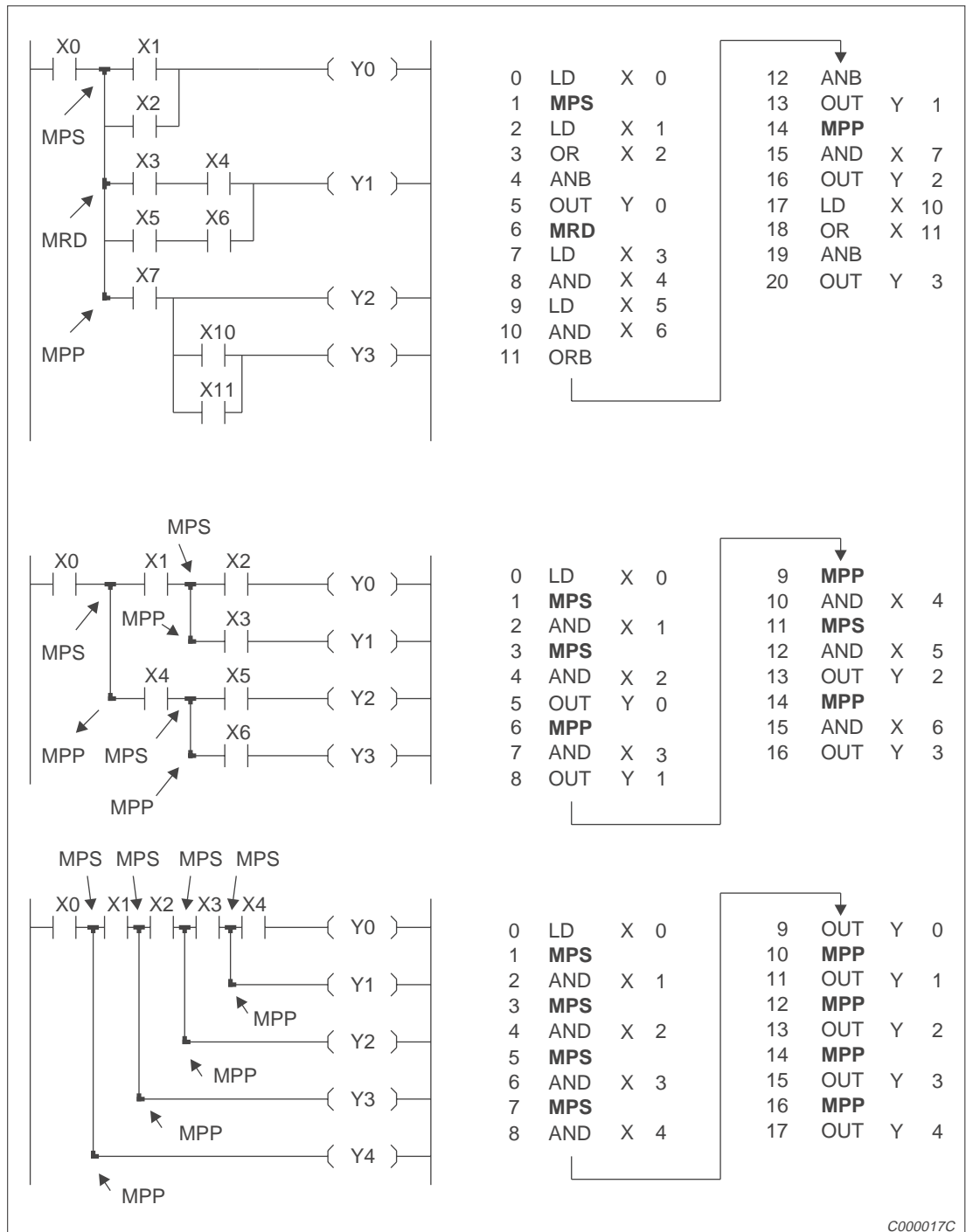
Beispiel ▾

Einsatz der Anweisungen MPS, MRD, MPP

**Abb. 4-12:** Programmierbeispiel zum Einsatz der Anweisungen MPS, MRD und MPP

- ① **MPS**
Das Zwischenergebnis (hier X0) in der 1. Verknüpfungsebene wird an der 1. Stelle im Verknüpfungsspeicher abgelegt.
- ② **MRD**
Vor Ausführung der nächsten Anweisung wird das Zwischenergebnis an der 1. Stelle des Verknüpfungsspeichers abgefragt.
- ③ **MPS**
Das Zwischenergebnis in der 2. Verknüpfungsebene wird an der 1. Stelle im Verknüpfungsspeicher abgelegt. Der bereits vorhandene Wert an der 1. Stelle rutscht an die 2. Stelle.
- ④ **MRD**
Vor Ausführung der nächsten Anweisung wird das Zwischenergebnis an der 1. Stelle des Verknüpfungsspeichers abgefragt.
- ⑤ **MRD**
Vor Ausführung der nächsten Anweisung wird das Zwischenergebnis an der 1. Stelle des Verknüpfungsspeichers abgefragt.
- ⑥ **MPP**
Vor Ausführung der nächsten Anweisung wird das Zwischenergebnis an der 1. Stelle im Verknüpfungsspeicher abgefragt. Die Operationen in der 2. Verknüpfungsebene werden abgeschlossen. Der Wert an der 1. Stelle im Verknüpfungsspeicher wird gelöscht. Der Wert der 2. Stelle rutscht zurück an die 1. Stelle.
- ⑦ **MPP**
Vor Ausführung der nächsten Anweisung wird das Zwischenergebnis an der 1. Stelle im Verknüpfungsspeicher abgefragt. Die Operationen in der 1. Verknüpfungsebene werden abgeschlossen, und der Verknüpfungsspeicher wird gelöscht.

△



C000017C

Abb. 4-13: Programmierbeispiel zum Einsatz der Anweisungen MPS, MRD und MPP

4.12 Setzen und Rücksetzen einer Kontrollbedingung (MC, MCR)

	MC			
	Master Control; Setzen einer Kontrollbedingung			
	CPU	FX1S	FX1N	FX2N
	●	●	●	●
MCR				
Master Control Reset; Rücksetzen einer Kontrollbedingung				
CPU	FX1S	FX1N	FX2N	
●	●	●	●	
Operanden	Programmschritte		Bemerkung	
MC: Y, M, keine Sondermerker	MC-Anweisung	3		
MCR: N	MCR-Anweisung	2		

Funktion

Durch Setzen (MC) oder Rücksetzen (MCR) einer Kontrollbedingung können einzelne Programmbereiche aktiviert bzw. deaktiviert werden. Die Funktion arbeitet also wie ein Hauptkontakt auf der linken Sammelschiene (Kontaktplanprogrammierung).

Beschreibung

- Mit der MC-Anweisung wird eine Kontrollbedingung zur Aktivierung eines bestimmten Programmbereiches gesetzt.
 - Welcher Programmbereich aktiviert werden soll, wird durch die Angabe der Programmverzweigungsadresse n: N0 bis N7 (Nesting-Adresse) festgelegt.
 - Die Angabe des Operanden Y oder M definiert einen Einschaltkontakt. Dieser Kontakt aktiviert den Programmbereich n, sobald die Eingangsbedingung für die MC-Anweisung erfüllt ist.
- Nach der Programmierung der MC-Anweisung müssen Sie immer eine LD- oder LDI-Anweisung programmieren (siehe Abs. 4.2)
- Die MCR-Anweisung setzt den MC-Kontakt zurück und stellt somit das Ende des Programmbereichs n dar.
- Ist die Eingangsbedingung nicht erfüllt, verändern sich die Zustände der Operanden zwischen MC und MCR wie folgt:
 - Bei remanenten Countern und Operanden, die in Zusammenhang mit den SET-, RST-Anweisungen programmiert werden, bleibt der Zustand erhalten (siehe Abs. 4.10.).
 - Alle Timer und Operanden, die in Zusammenhang mit der OUT-Anweisung programmiert wurden, werden zurückgesetzt.
- Innerhalb eines Programms können bis zu 8 Verzweigungsebenen aufgebaut werden. Die Verzweigungsebenen werden durch den Parameter „n“ gekennzeichnet.
- Was Sie beim Einsatz von mehreren MC- und MCR-Anweisungen innerhalb eines Programms beachten müssen, wird in einem der folgenden Beispiele erläutert.

HINWEIS

Durch den Einsatz der MC- bzw. MCR-Anweisung wird die Programmzykluszeit nicht verkürzt.

Beispiel ▾

Einsatz der Anweisungen MC, MCR

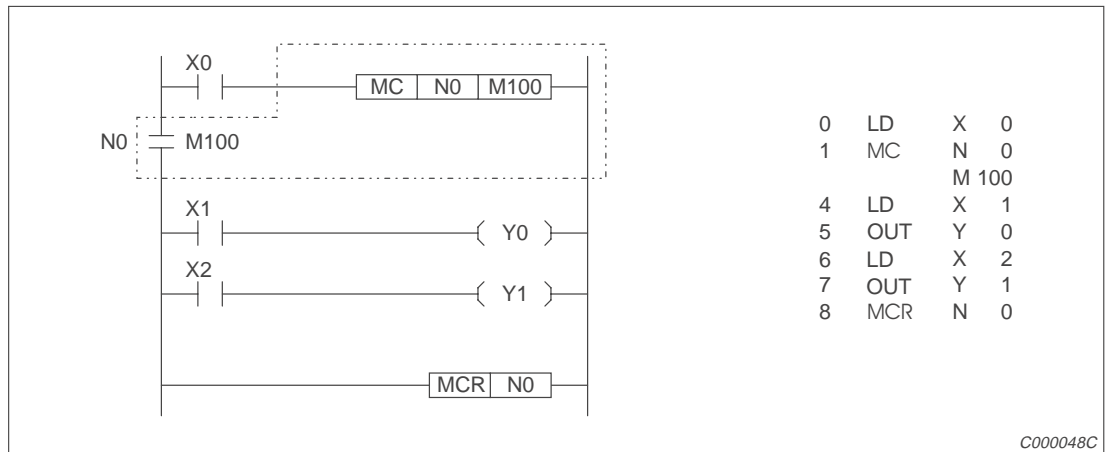


Abb. 4-14: Programmierbeispiel zum Einsatz der Anweisungen MC und MCR

Doppelbelegungen von Operanden innerhalb und außerhalb eines Master-Control-Bereichs führen auch bei inaktivem Master-Control-Bereich zu den in Abschnitt 4.3 (Doppelbelegung von Ausgängen) beschriebenen Problemen.

Sobald die Eingangsbedingung für die MC-Anweisung erfüllt ist, wird der Merkerkontakt M100 (Programmverzweigungsadresse N0) durchgeschaltet. Alle Strompfade zwischen der MC- und der MCR-Anweisung sind jetzt aktiviert. Der Signalzustand des Ausgangs Y0 bzw. Y1 ist dann nur noch vom Signalzustand des Eingangs X1 bzw. X2 abhängig. △

Einsatz von mehreren MC- und MCR-Anweisungen innerhalb eines Programms

Bei der Programmierung von mehreren MC- und MCR-Anweisungen innerhalb eines Programms müssen Sie darauf achten,

- daß die **erste MC**-Anweisung mit der **niedrigsten** Programmverzweigungsadresse N beginnt und
- daß die **erste MCR**-Anweisung mit der **höchsten** Programmverzweigungsadresse N beginnt.

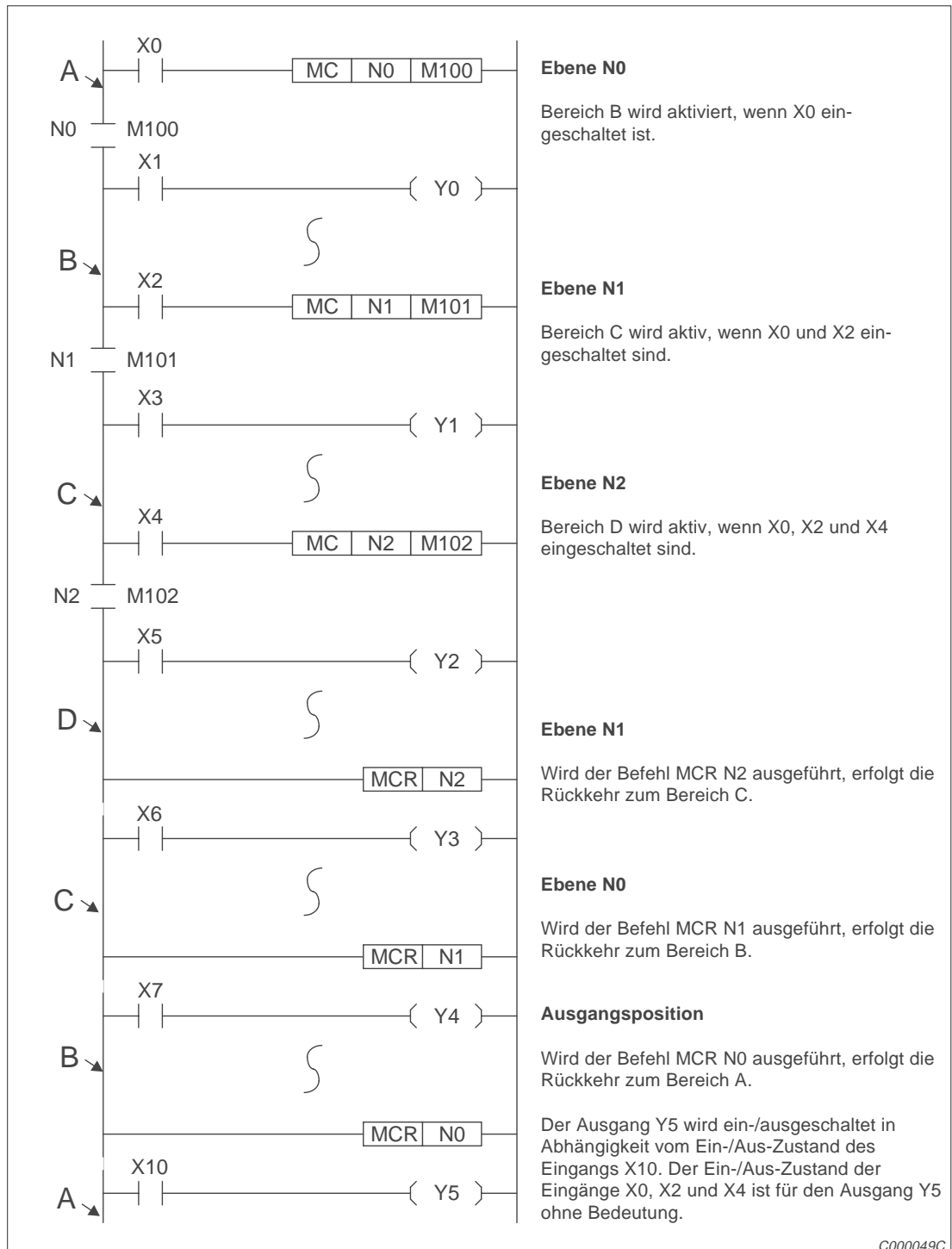



Abb. 4-15: Programmierbeispiel zum Einsatz von mehreren MC- und MCR-Anweisungen innerhalb eines Programms

4.13 Setzen und Rücksetzen von Operanden (SET, RST)

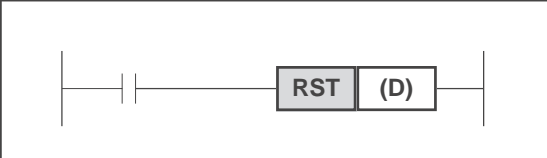
	SET			
	Setzen; Operanden setzen			
	CPU	FX1S	FX1N	FX2N
		●	●	●
Operanden	Programmschritte			Bemerkung
Y, M, S	Y, M	1		
	S, Sondermerker	2		

Funktion

Die Signalzustände von Operanden lassen sich mit den Anweisungen SET (Setzen) direkt festlegen.

Beschreibung

- Mit der SET-Anweisung können Sie einen Operanden Y, M oder S auf den Signalzustand „1“ setzen.
 - Sobald die Eingangsbedingung (Signal „1“) für die SET-Anweisung ansteht, wird der zugehörige Operand gesetzt.
 - Auch wenn die Eingangsbedingung für die SET-Anweisung nicht mehr ansteht, bleibt der zugehörige Operand gesetzt.

	RST				
	Rücksetzen; Operanden rücksetzen				
	CPU	FX1S	FX1N	FX2N	
		●	●	●	
Operanden	Programmschritte			Bemerkung	
Y, M, S, D, V, Z, T, C	Y, M	1	S, T, C	2	
	D, V, Z, Sonderregister			3	

Funktion

Die Signalzustände von Operanden lassen sich mit den Anweisungen RST (Rücksetzen) direkt festlegen.

- Mit der RST-Anweisung können Sie den zugehörigen Operanden zurücksetzen. Dies bedeutet:
 - Es werden Ausgänge Y, Merker M und Schrittstatusoperanden S ausgeschaltet (Signalzustand „0“).
 - Iswerte von Timern und Countern sowie Inhalte von Registern D, V und Z werden auf 0 zurückgesetzt.
 - Sobald die Eingangsbedingung (Signal „1“) für die RST-Anweisung ansteht, wird der zugehörige Operand zurückgesetzt.
 - Auch wenn die Eingangsbedingung für die RST-Anweisung nicht mehr ansteht, bleibt der zugehörige Operand zurückgesetzt.

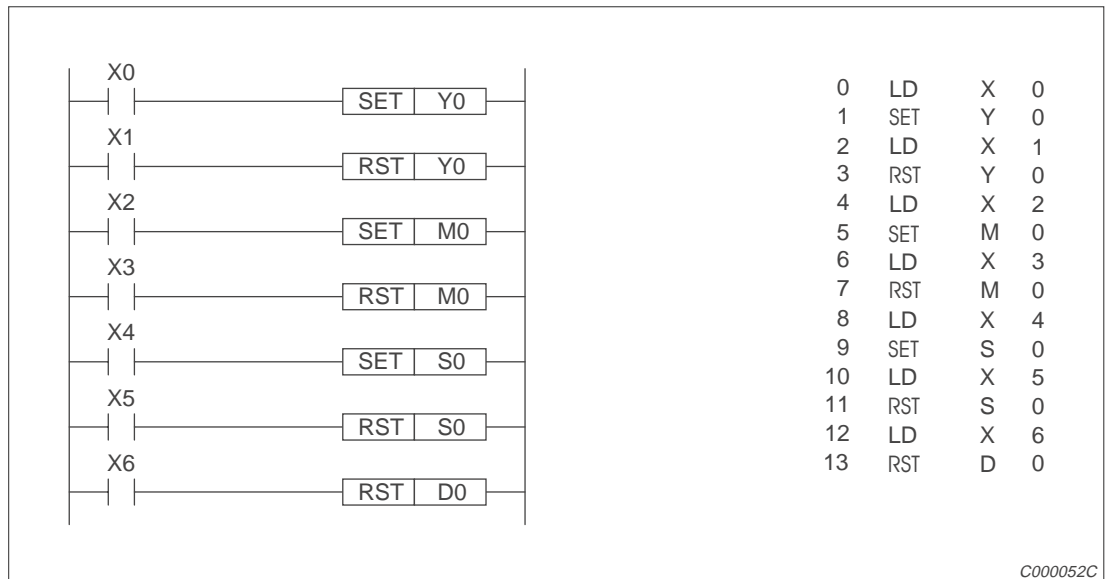


Abb. 4-16: Programmierbeispiel für den Einsatz der Anweisungen SET und RST

Beispiel ▾

Rücksetzen eines 16-Bit-Counters mit Hilfe der RST-Anweisung

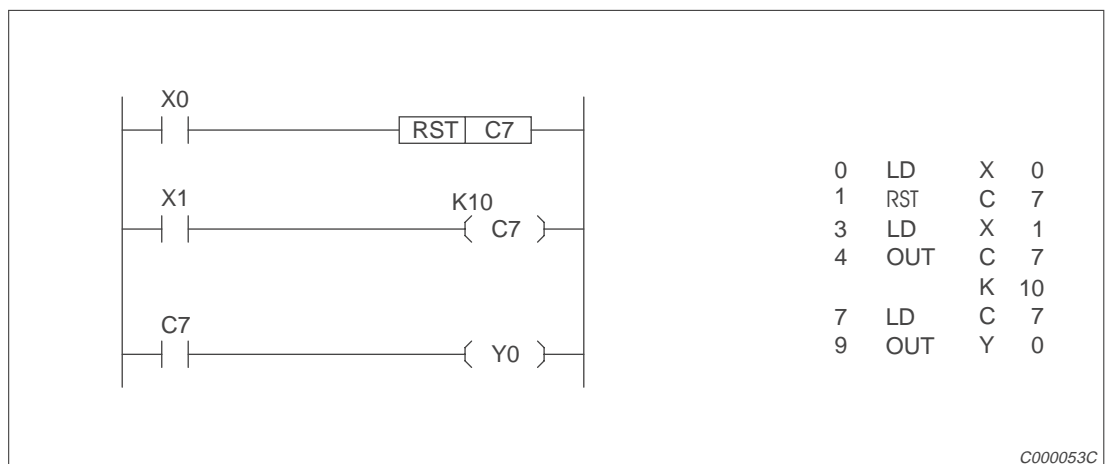


Abb. 4-17: Programmierbeispiel zum Rücksetzen eines 16-Bit-Counters mit Hilfe der RST-Anweisung

Der Ausgangskontakt Y0 wird aktiviert, wenn der Sollwert K10 erreicht ist. Sobald der Eingang X0 eingeschaltet ist, wird der Ausgangskontakt Y0 zurückgesetzt und der Istwert des Zählers C7 auf 0 zurückgesetzt. △

4.14 Erzeugen eines einmaligen Impulses (PLS, PLF)

	PLS		
	Impulserzeugung; Erzeugen eines einmaligen Impulses bei ansteigender Flanke		
	CPU	FX1S ●	FX1N ●
	PLF		
	Impulserzeugung; Erzeugung eines einmaligen Impulses bei abfallender Flanke		
	CPU	FX1S ●	FX1N ●
Operanden	Programmschritte		Bemerkung
Y, M	PLS-Anweisung	2	
	PLF-Anweisung	2	

Funktion

Erzeugen eines einmaligen Impulses - Flankenerkennung -, unabhängig von der Dauer des anstehenden Eingangssignals, zum Einschalten des zugehörigen Operanden. Der Operand bleibt für die Dauer eines Programmzyklus eingeschaltet.

Beschreibung

- Die PLS- und PLF-Anweisungen können im Zusammenhang mit Merkern M und digitalen Ausgängen Y benutzt werden. Die Anweisungen erzeugen einen gleichbleibenden Impuls, unabhängig von der Dauer des anstehenden Eingangssignals.
- Nach Ausführung einer PLS- oder PLF-Anweisung steht das Signal des zugehörigen Operanden (Y oder M) während der Dauer eines Programmzyklus an.
- Die PLS-Anweisung erzeugt einen einmaligen Impuls bei ansteigender Flanke des Eingangssignals.
- Die PLF-Anweisung erzeugt einen einmaligen Impuls bei abfallender Flanke des Eingangssignals.

HINWEIS

| Sondermerker dürfen nicht mit einer PLS- oder PLF-Anweisung aktiviert werden.

Beispiel ▾ Einsatz der Anweisungen PLS, PLF

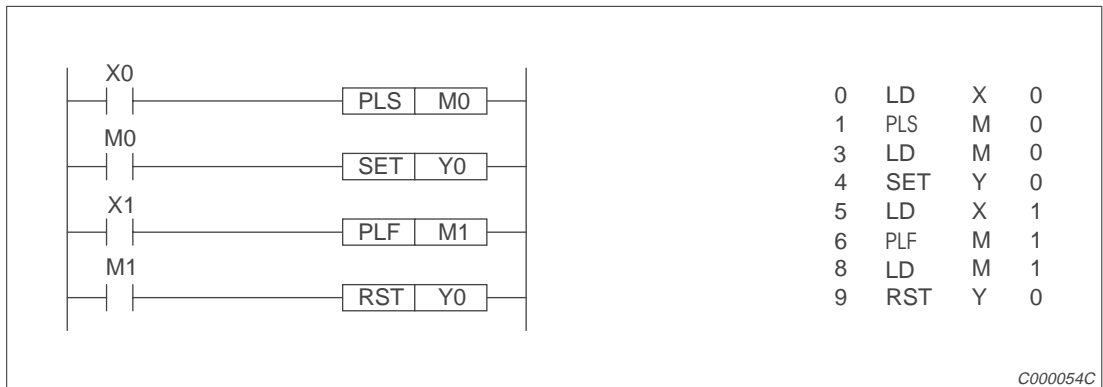


Abb. 4-18: Programmierbeispiel zum Einsatz der Anweisungen PLS und PLF

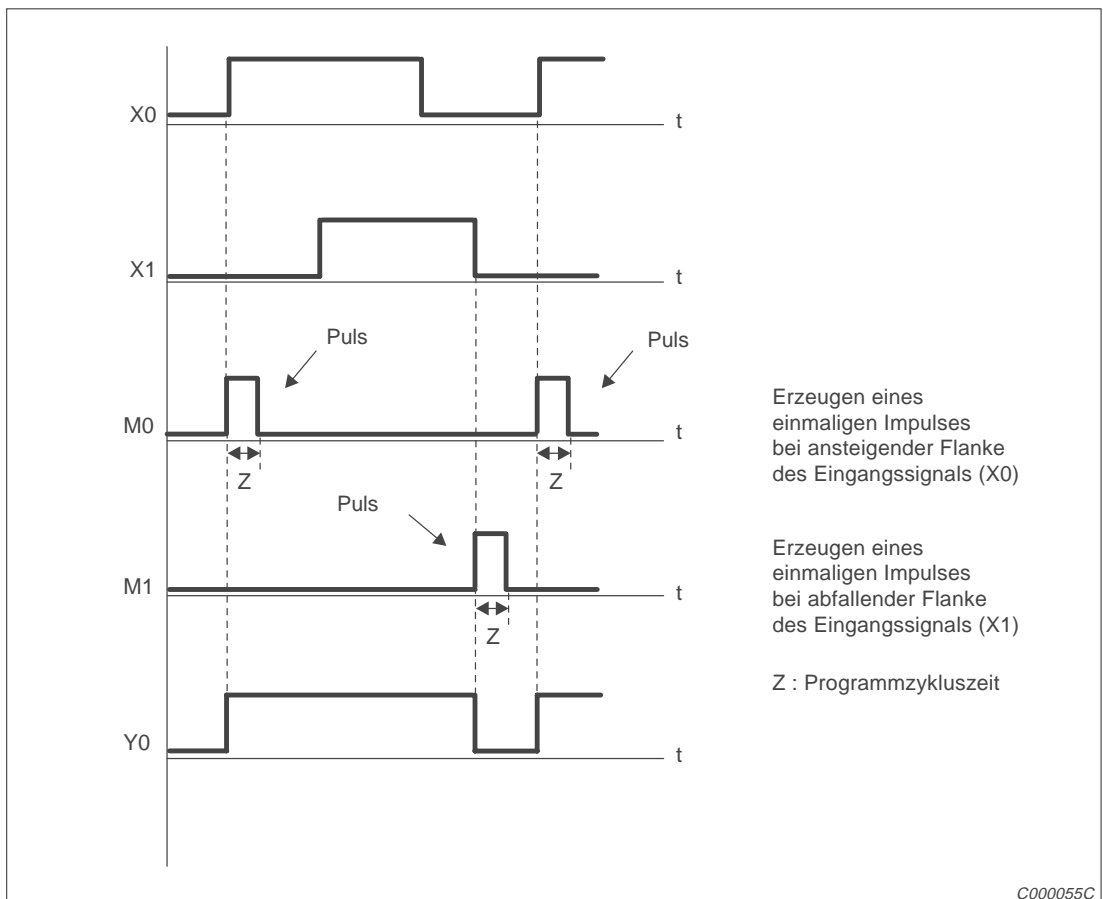


Abb. 4-19: Programmierbeispiel, Darstellung der Eingangssignalverarbeitung und der Impulserzeugung

Beim Wechseln des Eingangssignals am Eingang X0 von „0“ auf „1“ (ansteigende Flanke) erhält der Merker M0 durch die PLS-Anweisung einen Impuls. Mit diesem Impuls wird über den Merkerkontakt M0 der Ausgang Y0 gesetzt. Erst wenn am Eingang X1 ein Eingangssignalwechsel von „1“ auf „0“ (abfallende Flanke) auftritt, wird der Ausgang Y0 wieder zurückgesetzt.



4.15 Inversion von Verarbeitungsergebnissen (INV)

	INV			
	Inversion; Inversion von Verarbeitungsergebnissen			
	CPU	FX1S	FX1N	FX2N
		●	●	●
Operanden	Programmschritte			Bemerkung
—	INV-Anweisung	1		

Funktion

Umkehren des Signalzustandes von Verarbeitungsergebnissen

Beschreibung

Die INV-Anweisung invertiert den Signalzustand des Ergebnisses vorstehender Anweisung.

- Lautet das Verarbeitungsergebnis 1, wird es nach der Inversion 0.
- Lautet das Verarbeitungsergebnis 0, wird es nach der Inversion 1.
- Die INV-Anweisung kann wie die AND- und ANI-Anweisungen verwendet werden.

HINWEISE

Die INV-Anweisung kann zur Signalumkehrung des Ergebnisses einer komplexen Schaltung verwendet werden.

Die INV-Anweisung kann zur Signalumkehr der Ergebnisse der gepulsten Anweisungen LDP, LDF, ANP etc. verwendet werden.

Beispiel ▾

Einsatz der INV-Anweisung

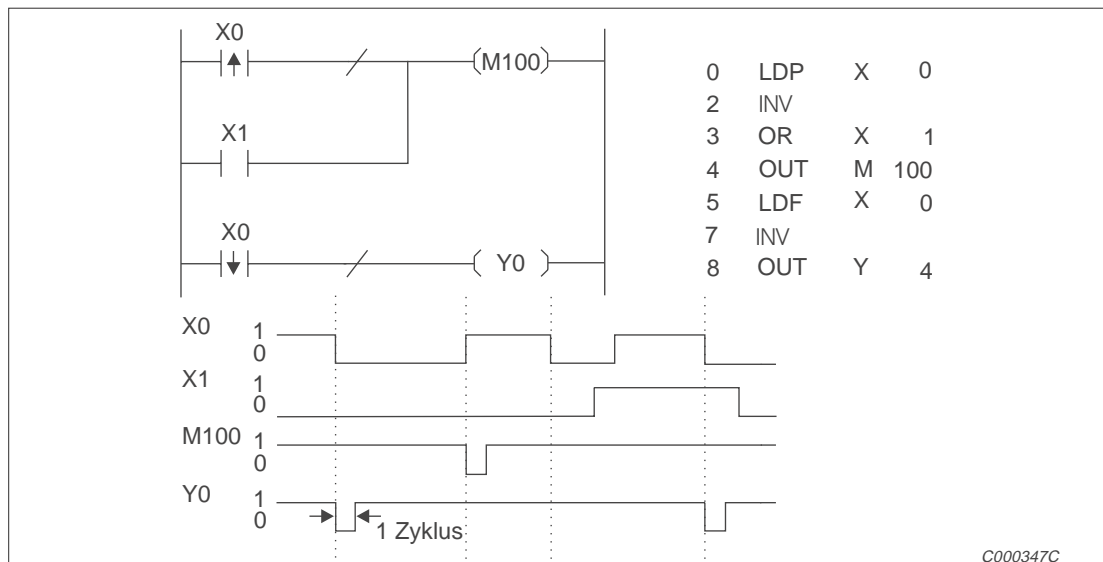


Abb. 4-20: Programmierbeispiel zum Einsatz der INV-Anweisung

Der Merker M100 wird mit positiver Flanke von X0 zurückgesetzt.

Der Ausgang Y0 wird mit abfallender Flanke von X0 zurückgesetzt.



4.16 Leerzeile im Programm (NOP)

		NOP			
		Leerzeile; Leerzeile im Programm ohne logische Funktion			
Operanden		CPU	FX1S	FX1N	FX2N
		—			●
Operanden		Programmschritte			Bemerkung
—		NOP-Anweisung			1

Funktion

Es wird eine Leerzeile ohne eine logische Funktion erzeugt, die später durch weitere Anweisungen in einem noch nicht fertiggestellten Programm aufgefüllt werden kann.

Beschreibung

- Nach Abschluß der Programmierfolge sollten NOP-Befehle gelöscht werden, da ansonsten die Programmzykluszeit unnötig verlängert wird.
- Die Anzahl der NOP-Befehle ist nicht begrenzt.
- Beim Löschen des gesamten Programms werden sämtliche Anweisungen durch NOP-Anweisungen überschrieben.
- Nachträgliches Einfügen von NOP-Anweisungen mit einem Handprogrammiergerät sollten Sie mit der INSERT-Funktion durchführen.

HINWEIS

Werden die Anweisungen LD, LDI, ANB oder ORB durch eine NOP-Anweisung ersetzt, kann sich der logische Schaltungsaufbau wesentlich verändern. Hierdurch kann es zu Programmablauf Fehlern kommen.

Beispiel ▾ Einsatz der Anweisung NOP

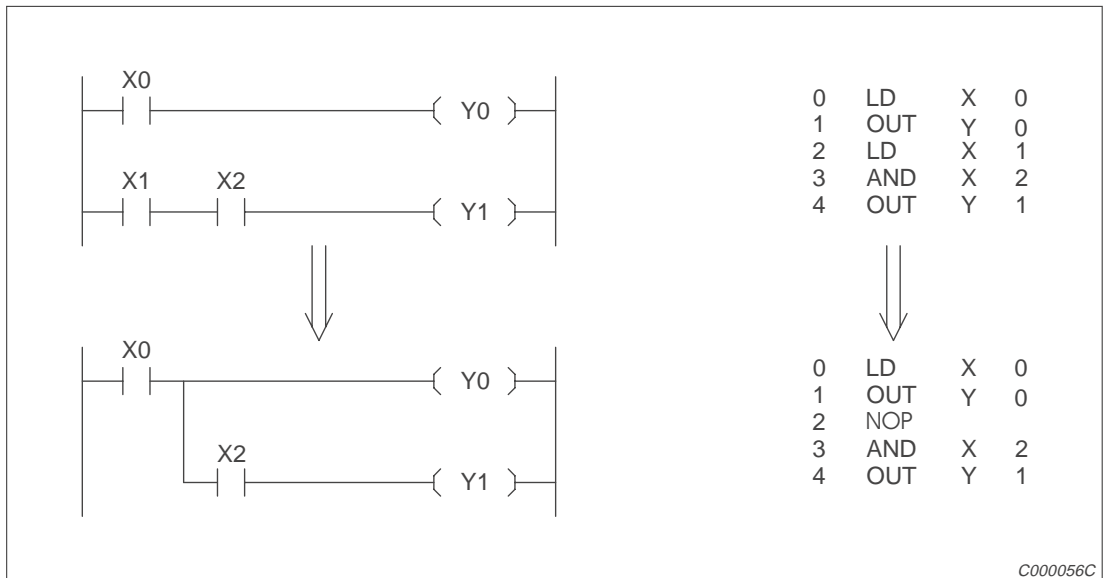
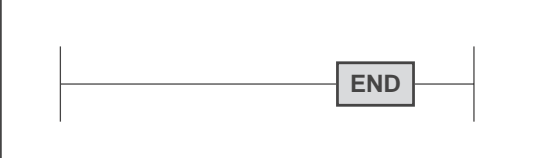


Abb. 4-21: Programmierbeispiel zum Einsatz der Anweisung NOP

Der Eingang X1 wird durch eine NOP-Anweisung ersetzt. Der logische Schaltungsaufbau hat sich hierdurch geändert. △

4.17 SPS-Programmende (END)

	END			
	Ende; SPS-Programmende Sprung zum Programmanfang			
	CPU	FX1S	FX1N	FX2N
		●	●	●
Operanden	Programmschritte			Bemerkung
—	END-Anweisung	1		

Funktion

Abschließen eines SPS-Programms und Sprung zum Programmanfang (Schritt 0)

Beschreibung

- Jedes SPS-Programm müssen Sie mit einer END-Anweisung abschließen.
- Wurde eine END-Anweisung programmiert, wird an dieser Stelle die Programmabarbeitung beendet. Nachfolgende Programmbereiche werden nicht mehr berücksichtigt. Nach Abarbeitung der END-Anweisung erfolgt die Ausgangsbearbeitung. Die Programmabarbeitung springt dann zum Programmanfang (Schritt 0) zurück.
- Um zur schrittweisen Programmüberprüfung einzelne Programmabschnitte zu erzeugen, kann die END-Anweisung auch innerhalb des Programms eingesetzt werden. Die Anweisungen nach der END-Anweisung werden bei der Überprüfung nicht berücksichtigt. Diese „eingeschobenen“ END-Anweisungen müssen Sie anschließend wieder löschen.

HINWEIS

Nach Ausführung der END-Anweisung findet eine Auffrischung des Watch-Dog-Timers und der Image-Register statt.

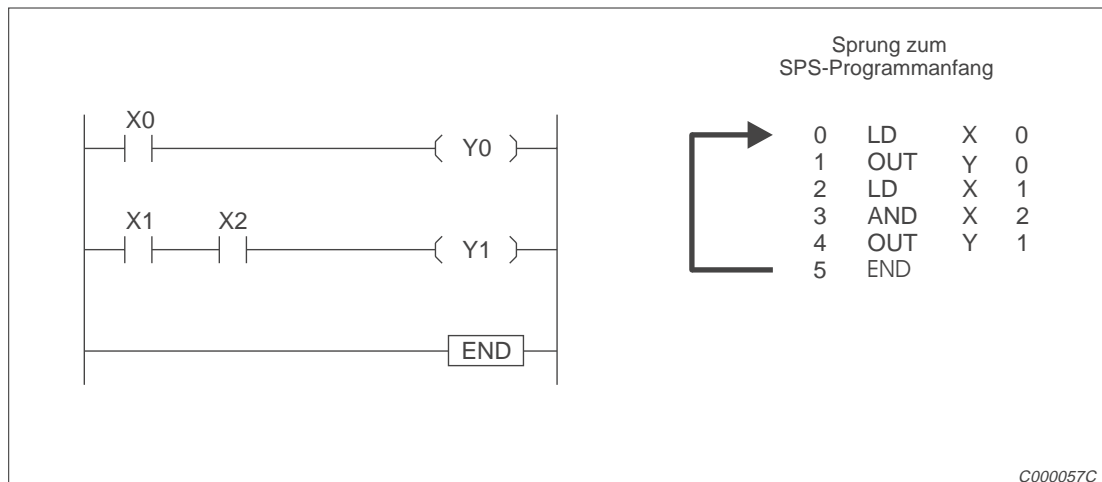


Abb. 4-22: Programmierbeispiel zum Einsatz der END-Anweisung

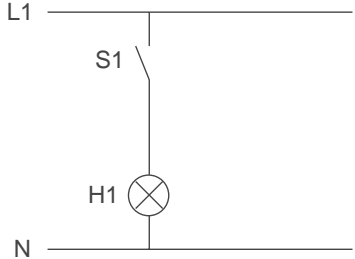
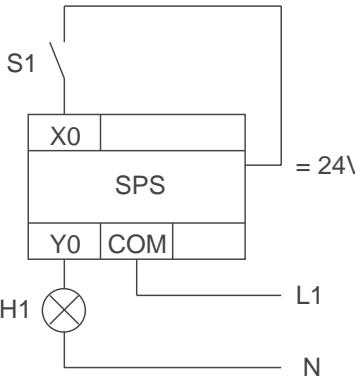
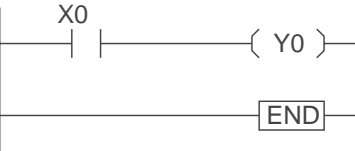
4.18 Programmbeispiele

Der folgende Abschnitt zeigt einige einfache Beispiele zur Anwendung des Grundbefehlssatzes. Die Beispiele können direkt programmiert und ausgeführt werden und dienen der Vertiefung des im ersten Teil dieses Handbuches erlangten Wissens.

- Abfrage eines Eingangs (Öffner und Schließer)
- Reihenschaltung von Eingängen
- Parallelschaltung von Eingängen
- Selbsthaltung eines Ausgangs
- Einschaltverzögerung
- Ausschaltverzögerung
- Aufwärtszähler

4.18.1 Abfrage eines Eingangs

Betätigter Schließer

Beispiel	Stromlaufplan
<p>Bei Betätigen des Schließers S1 soll der Melder H1 leuchten.</p>	 <p style="text-align: right;">C000007G</p>
Zuordnungsliste	Beschaltung der SPS
<p>Schließer: S1 X0 Melder: H1 Y0</p>	 <p style="text-align: right;">C000008G</p>
Anweisungsliste	Kontaktplan
<p>0 LD X0 1 OUT Y0 2 END</p>	 <p style="text-align: right;">C000009G</p>
Anmerkung	
<p>Der Ausgang Y0 führt das Signal „1“, wenn an Eingang X0 das Signal „1“ ansteht.</p>	

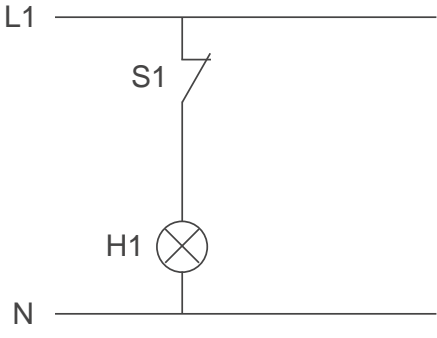
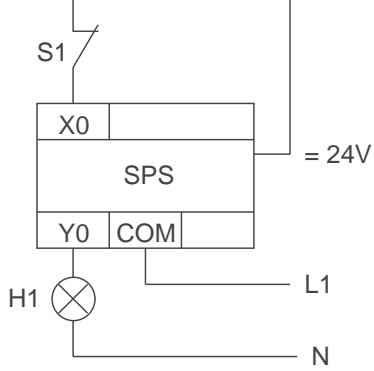

Tab. 4-4: Beispiel für einen betätigten Schließer



HINWEIS

Benutzen Sie als Befehlsgeber zum Einschalten von Betriebszuständen immer Schließer, damit die SPS bei einem Erdschluß die Steuerungsanlage abschalten kann.

Nichtbetätigter Öffner

Beispiel	Stromlaufplan
<p>Bei Nichtbetätigen des Öffners S1 soll der Melder H1 leuchten.</p>	 <p style="text-align: right;">C000010G</p>
Zuordnungsliste	Beschaltung der SPS
<p>Öffner: S1 X0 Melder: H1 Y0</p>	 <p style="text-align: right;">C000011G</p>
Anweisungsliste	Kontaktplan
<pre> 0 LD X0 1 OUT Y0 2 END </pre>	 <p style="text-align: right;">C000012G</p>
Anmerkung	
<p>Der Ausgang Y0 führt das Signal „1“, wenn an Eingang X0 das „1“-Signal ansteht. Der Öffner S1 wird im Programm auf den Signalzustand „1“ abgefragt, damit der Ausgang Y0 bei einer Betätigung des Öffners S1 ein „0“-Signal aufweist.</p>	

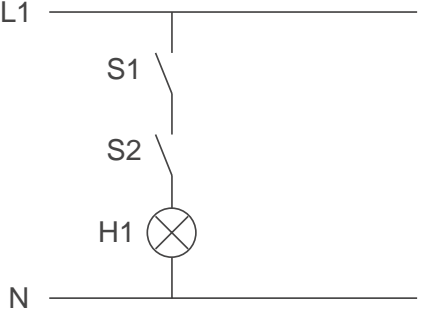
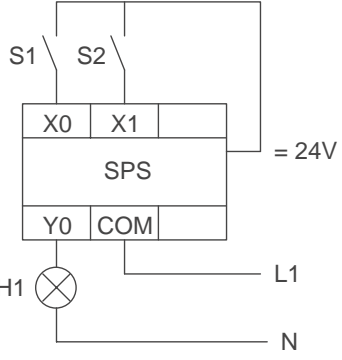
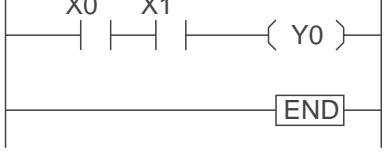
Tab. 4-5: Beispiel für einen nichtbetätigten Öffner



HINWEIS

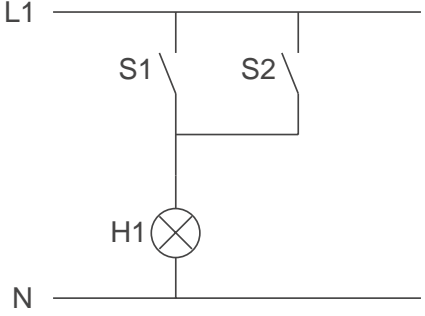
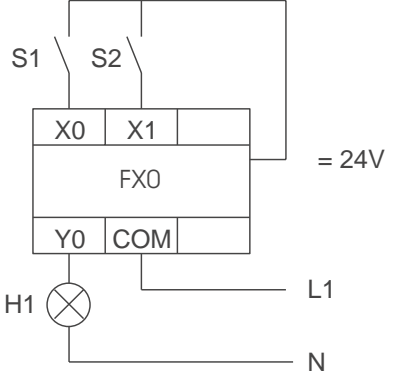
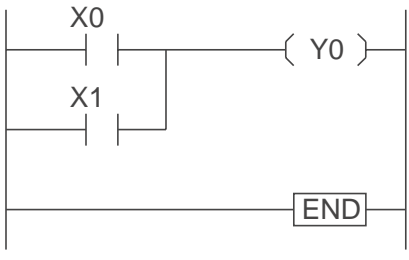
Benutzen Sie zum Ausschalten von Betriebszuständen als Befehlsgeber immer Öffner, damit die SPS bei einem Erdschluß die Steuerungsanlage abschalten kann.

Reihenschaltung

Beispiel	Stromlaufplan
<p>Bei Betätigen des Schließers S1 UND des Schließers S2 soll der Melder H1 leuchten.</p>	 <p style="text-align: right;">C00001</p>
Zuordnungsliste	Beschaltung der SPS
<p>Schließer: S1 X0 Schließer: S2 X1 Melder: H1 Y0</p>	 <p style="text-align: right;">C000017G</p>
Anweisungsliste	Kontaktplan
<p>0 LD X0 1 AND X1 2 OUT Y0 3 END</p>	 <p style="text-align: right;">C000018G</p>
Anmerkung	
<p>Der Ausgang Y0 führt das Signal „1“, wenn der Eingang X0 und der Eingang X1 ein „1“-Signal aufweisen.</p>	

Tab. 4-6: Beispiel für eine UND-Verknüpfung

Parallelschaltung

Beispiel	Stromlaufplan
<p>Bei Betätigen des Schließers S1 ODER des Schließers S2 soll der Melder H1 leuchten.</p>	 <p style="text-align: right;"><small>C000019G</small></p>
Zuordnungsliste	Beschriftung der SPS
<p>Schließer: S1 X0 Schließer: S2 X1 Melder: H1 Y0</p>	 <p style="text-align: right;"><small>C000020G</small></p>
Anweisungsliste	Kontaktplan
<p>0 LD X0 1 OR X1 2 OUT Y0 3 END</p>	 <p style="text-align: right;"><small>C000021G</small></p>
Anmerkung	
<p>Der Ausgang Y0 führt das Signal „1“, wenn mindestens ein Eingang X0 oder X1 den Signalzustand „1“ aufweist.</p>	

Tab. 4-7: Beispiel für eine ODER-Verknüpfung

Selbsthaltung (I)

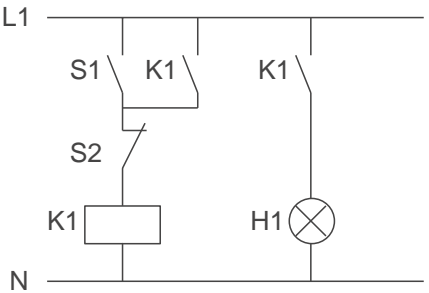
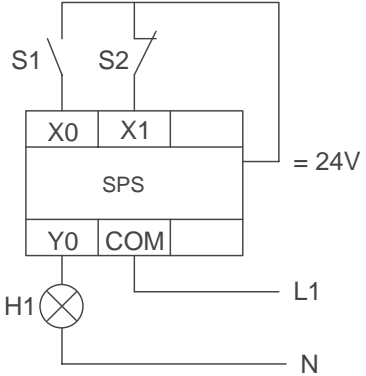
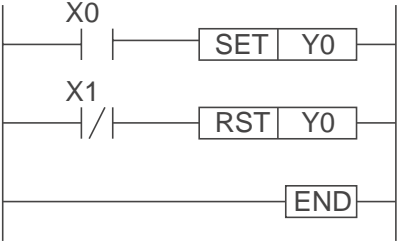
Setzen eines Ausgangs mit Selbsthaltung

<p>Beispiel</p> <p>Bei Betätigen des Schließers S1 soll der Melder H1 leuchten, auch wenn der Schließer S1 nicht mehr betätigt wird. Nach kurzzeitigen Betätigen des Öffners S2 soll der Melder H1 nicht mehr leuchten.</p>	<p>Stromlaufplan</p> <p style="text-align: right;">C000033G</p>
<p>Zuordnungsliste</p> <p>Schließer: S1 X0 Öffner: S2 X1 Melder: H1 Y0</p>	<p>Beschaltung der SPS</p> <p style="text-align: right;">C000034G</p>
<p>Anweisungsliste</p> <pre> 0 LD X0 1 OR Y0 2 AND X1 3 OUT Y0 4 END </pre>	<p>Kontaktplan</p> <p style="text-align: right;">C000036G</p>
<p>Anmerkung</p> <p>Der Ausgang Y0 wird eingeschaltet (Signalzustand „1“), wenn der Eingang X0 kurzzeitig durchgeschaltet wird (Schließer S1 betätigt). Der Ausgang Y0 wird ausgeschaltet (Signalzustand „0“), wenn der Eingang X1 kurzzeitig betätigt wird (Öffner S2 betätigt).</p>	

Tab. 4-8: Beispiel zum Setzen und Rücksetzen eines Ausgangs mit Selbsthaltung

Selbsthaltung (II)

Setzen und Rücksetzen eines Ausgangs mit SET-/RST-Anweisung

Beispiel	Stromlaufplan
<p>Bei Betätigen des Schließers S1 soll der Melder H1 leuchten, auch wenn der Schließer S1 nicht mehr betätigt wird. Nach kurzzeitigen Betätigen des Öffners S2 soll der Melder H1 nicht mehr leuchten.</p>	 <p style="text-align: right;"><small>C000033G</small></p>
Zuordnungsliste	Beschaltung der SPS
<p>Schließer: S1 X0 Öffner: S2 X1 Melder: H1 Y0</p>	 <p style="text-align: right;"><small>C000034G</small></p>
Anweisungsliste	Kontaktplan
<pre> 0 LD X0 1 SET Y0 2 LDI X1 3 RST Y0 4 END </pre>	 <p style="text-align: right;"><small>C000035G</small></p>
Anmerkung	
<p>Der Ausgang Y0 wird eingeschaltet (Signalzustand „1“), wenn der Eingang X0 kurzzeitig durchgeschaltet wird (Schließer S1 betätigt). Der Ausgang Y0 wird ausgeschaltet (Signalzustand „0“), wenn der Eingang X1 kurzzeitig betätigt wird (Öffner S2 betätigt).</p>	

Tab. 4-9: Beispiel zum Setzen (SET) und Rücksetzen (RST) eines Ausgangs

Einsatz eines Timers zur Einschaltverzögerung

Beispiel	Zuordnungsliste
Bei Betätigen des Schließers S1 soll der Melder H1 nach $t = 5\text{ s}$ leuchten.	Schließer: S1 X0 Melder: H1 Y0 Timer: T0 100 ms
Zeitdiagramm	Beschaltung der SPS
Anweisungsliste	Kontaktplan
<pre> 0 LD X0 1 OUT T0 K50 4 LD T0 5 OUT Y0 6 END </pre>	
Anmerkung	
Wenn der Eingang X0 den Signalzustand „1“ aufweist, beginnt die Zeit abzulaufen. Nach Ablauf der programmierten Zeit $t = 5\text{ s}$ wird der Ausgang Y0 auf den Signalzustand „1“ geschaltet. Der Timer T0 fällt in den Ruhezustand „0“ zurück, sobald der Eingang X0 den Signalzustand „0“ aufweist.	

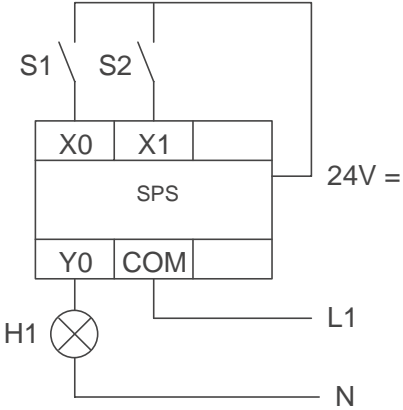
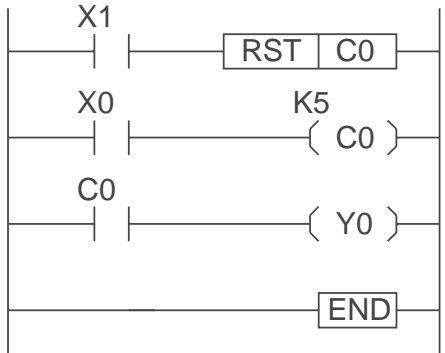
Tab. 4-10: Beispiel zum Einsatz eines Timers für eine Einschaltverzögerung

Einsatz eines Timers zur Ausschaltverzögerung

Beispiel	Zuordnungsliste
Bei Betätigen des Schließers S1 soll der Melder H1 sofort leuchten. Der Melder H1 soll jedoch $t = 5\text{ s}$ länger leuchten, als der Schließer S1 betätigt wird.	Schließer: S1 X0 Melder: H1 Y0 Timer: T0 100 ms
Zeitdiagramm	Beschaltung der SPS
Anweisungsliste	Kontaktplan
<pre> 0 LD X0 1 OR Y0 2 ANI T0 3 OUT Y0 4 LD Y0 5 ANI X0 5 OUT T0 K50 9 END </pre>	
Anmerkung	
Wenn der Eingang X0 den Signalzustand „1“ aufweist, schaltet der Ausgang Y0 in „Selbsthaltung“. Sobald der Eingang X0 auf das „0“-Signal zurückfällt, läuft die programmierte Zeit. Nach Ablauf der eingestellten Zeit $t = 5\text{ s}$ wird der Ausgang Y0 auf den Signalzustand „0“ zurückgesetzt.	

Tab. 4-11: Beispiel für den Einsatz eines Timers zur Ausschaltverzögerung

Einsatz eines Aufwärtszählers

<p>Beispiel</p> <p>Nach 5 Betätigungen des Schließers S1 soll der Melder H1 aufleuchten. Mit dem Schließer S2 soll der Zähler wieder auf den Anfangszustand zurückgesetzt werden und der Melder H1 verlöschen.</p>	<p>—</p> <p>—</p>
<p>Zuordnungsliste</p> <p>Schließer: S1 X0</p> <p>Schließer: S2 X1</p> <p>Melder: H1 Y0</p> <p>Zähler: C0</p>	<p>Beschaltung der SPS</p> 
<p>Anweisungsliste</p> <p>0 LD X1</p> <p>1 RST C0</p> <p>3 LD X0</p> <p>4 OUT C0 K5</p> <p>7 LD C0</p> <p>8 OUT Y0</p> <p>9 END</p>	<p>Kontaktplan</p> 
<p>Anmerkung</p> <p>Bei jedem Ansteuern des Zählers C0 mit einem „1“-Signal wird der Zähleristwert um den Wert 1 erhöht. Nachdem der Zähleristwert den Zählersollwert 5 erreicht hat, schaltet der Zähler den Ausgang Y0 auf den Signalzustand „1“. Durch ein „1“-Signal am Eingang X1 wird der Zähler wieder auf den Signalzustand „0“ zurückgesetzt.</p>	

Tab. 4-12: Programmierbeispiel zum Einsatz eines Aufwärtszählers

5 STL-Anweisung

5.1 Allgemeine Hinweise

Die STL-Anweisung ist eine elementare SPS-Anweisung zur einheitlichen Programmierung von Steuerungsabläufen. Die STL-Anweisung wird in Verbindung mit einem Schrittstatus eingesetzt und ermöglicht die komfortable Programmierung von Schrittsteuerungen.

Aufwendige Programme für einfache Start-/Stopp-Sequenzen entfallen bei der Programmierung, so daß auch der Programmieranfänger die Steuerung effektiv nutzen kann. Sie können damit den Programmieraufwand für derartige Sequenzen erheblich einschränken.

Die STL-Anweisung wird in Zusammenhang mit dem Schrittstatusoperanden S programmiert. Je nach verwendetem CPU-Typ stehen bis zu 1000 Schrittstatusoperanden im Bereich von S0 bis S999 zur Verfügung, wobei den Operanden S0 bis S9 feste Funktionen zugeordnet sind.

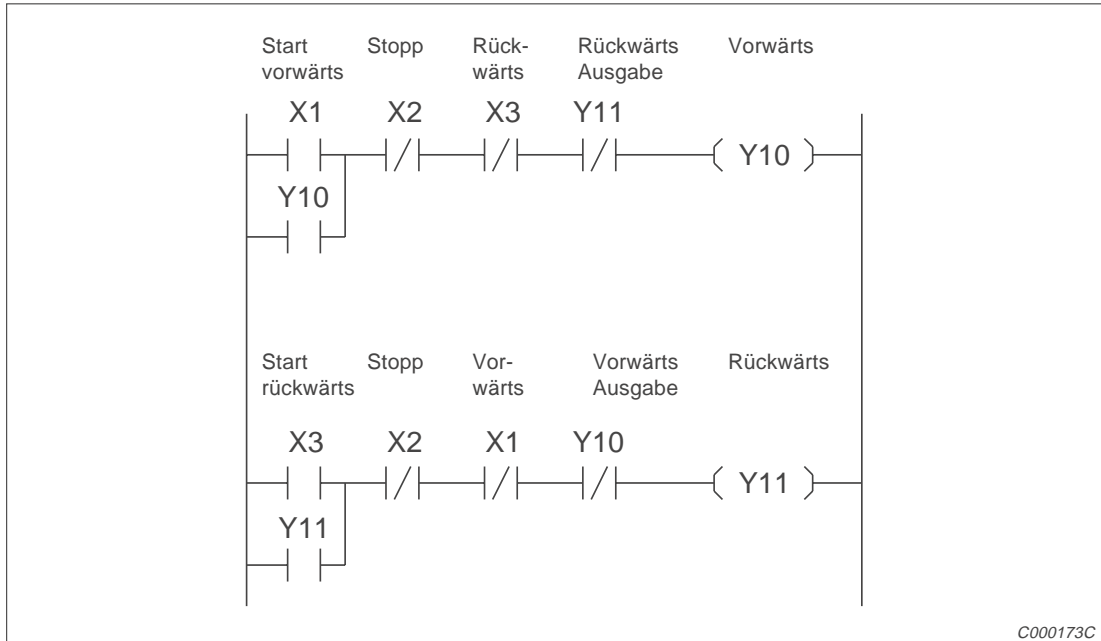
Bedeutung der Schrittstatusoperanden S0 bis S9

Die Schrittstatusoperanden S0 bis S9 sind Initialisierungsoperanden, mit denen verschiedene Schrittabläufe innerhalb eines STL-Programms erstellt werden können, um z.B. unterschiedliche Betriebsabläufe (Hand-/Automatikbetrieb, Nullpunktfahrt usw.) zu realisieren. Dies gilt insbesondere, wenn die Applikationsanweisung IST genutzt wird.

Wenn keine gesonderten Abläufe für Hand-, Automatikbetrieb und Nullpunktfahrt vorgesehen sind, können die Operanden S0 bis S9 als „normale“ Operanden ohne Sonderfunktionen genutzt werden.

5.1.1 Anwendungsbeispiel zum Einsatz der STL-Anweisung

Die herkömmliche Programmiermethode mittels Kontaktplan besteht darin, daß ein Ausgang von einem bestimmten, ihm zugeordneten Eingangskontakt (z.B. externer mechanischer Schalter) angesprochen wird und dieser Eingangskontakt parallel oder in Reihe zur Steuerungsaufgabe liegt. Zur Sicherung des Arbeitsprozesses gegen ungewollte Steuerungsabläufe und damit zusammenhängende Fehlfunktionen muß ein solches Programm umfangreiche Verriegelungsmaßnahmen enthalten.



C000173C

Abb. 5-2: Anwendungsbeispiel mit Verriegelungskontakten

Bei der Verwendung von Schrittsteueranweisungen können die gezeigten Verriegelungskontakte entfallen, da die Steuersignale, wie z.B. „aufwärts“, „abwärts“ usw., über das Programm unter Berücksichtigung bestimmter Grenzwerte erfolgen.

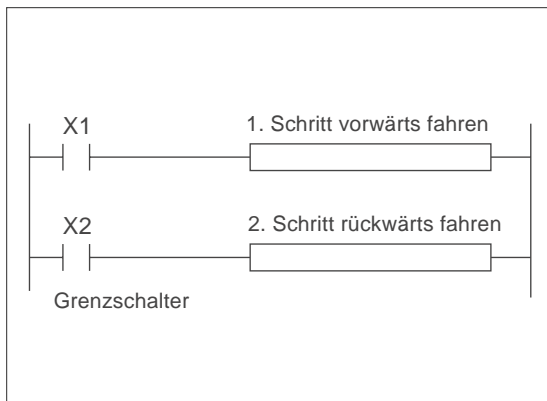


Abb. 5-1: Anwendungsbeispiel ohne Verriegelungskontakte

C000129C

5.1.2 Schematischer Ablauf einer Schrittsteuerung

Anhand einer kurzen Prozeßfolge wird als Beispiel eine Schrittsteuerung mit vier Arbeitsschritten beschrieben. Der vierte Schritt beendet die Schrittsteuerung.

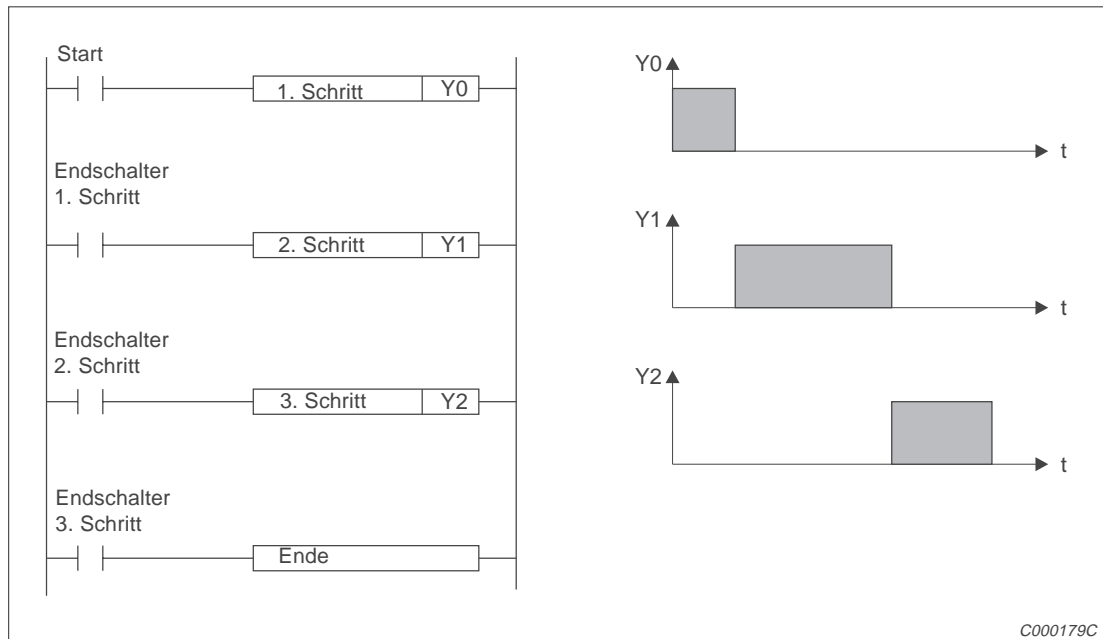


Abb. 5-3: Schematischer Ablauf einer Schrittsteuerung

In der Abb. 5-3 ist zu sehen, daß der 2. Arbeitsgang eingeschaltet wird, sobald der 1. Arbeitsschritt beendet ist und der zugehörige Endschalter eingeschaltet wird. Dies beinhaltet, daß auch alle Zustände der Operanden innerhalb des 1. Arbeitsschrittes zurückgesetzt werden.

Das Ende des 2. Arbeitsschrittes bedeutet gleichzeitig den Start des 3. Arbeitsschrittes. Mit dem Einschalten des 3. Endschalters wird das Ende der Schrittfolge (4. Arbeitsgang) erreicht.

5.1.3 Darstellung einer Ablaufsteuerung in einem Flußdiagramm

In der folgenden Abb. ist die gleiche Ablaufsteuerung in einem Flußdiagramm (IEC-Standard) dargestellt. In einem Flußdiagramm ist die Darstellung einer Ablaufsteuerung erst einmal unabhängig von der späteren Realisierung in einem SPS-Programm.

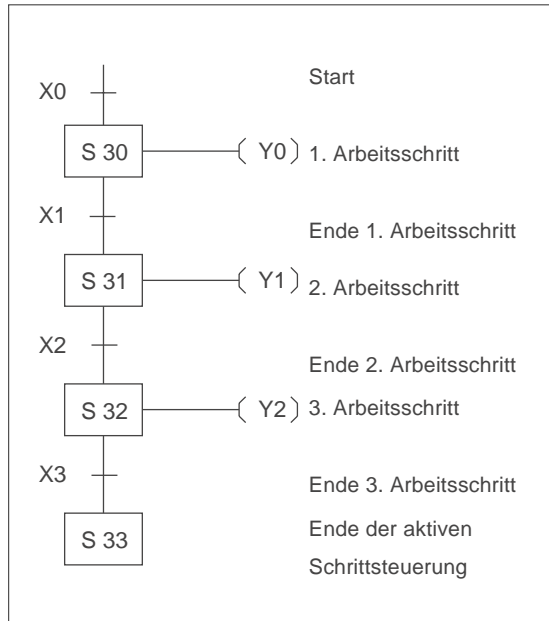


Abb. 5-4:
Flußdiagramm des
Anwendungsbeispiels

C000148C

5.2 STL-Anweisung programmieren

	STL		RET	
	Schrittstatus aktivieren/deaktivieren			
	CPU	FX1S	FX1N	FX2N
	●	●	●	
Operanden	Puls-Anweisung (P)	Verarbeitung	Programmschritte	
S0 – S999; Die Adressbereiche sind abhängig von der verwendeten MELSEC SPS (siehe Tabelle 5-1).		16 Bit	32 Bit	STL
				RET
				1
				1

Steuerung	Anzahl der Operanden	Adressbereich
FX1S	128	S0 bis S127
F1NX	1000	S0 bis S999
FX2N	1000	S0 bis S999

Tab. 5-1: Adressbereiche

Funktion

Programmierung von Ablaufsteuerungen

Beschreibung

- Die STL-Anweisung wird zusammen mit dem Schrittstatusoperanden S eingesetzt. Der Schrittstatusoperand S kann mit den folgenden Anweisungen des Grundbefehlssatzes programmiert werden: LD, LDI, AND, ANI, OR, ORI, OUT, SET, RST.
- Bei der FX2N-Serie ist die Verwendung von gepulsten Anweisungen (LDP, LDF etc.) und gepulsten Merkern (M2800 - M3071) möglich.
- In einem Programm ohne Schrittsteuerung können die Schrittstatusoperanden S auch als herkömmliche Merker verwendet werden.
- Am Anfang eines STL-Programmbereiches (Schrittstatus) muß jeder einzelne Schrittstatusoperand mit der SET-Anweisung gesetzt werden.
- Innerhalb des Kontaktplanes tritt der STL-Kontakt an der linken Sammelschiene auf und kann daher als „Hauptkontakt“ angesehen werden.
- Der einer STL-Anweisung folgende Strompfad kann erst bearbeitet werden, wenn der STL-Kontakt gesetzt ist.
- Sobald der STL-Kontakt zurückgesetzt ist, kann der nachstehende Strompfad nicht mehr bearbeitet werden.
- Mit der RET-Anweisung wird der gesamte STL-Programmbereich (Schrittstatus) beendet.

HINWEISE

- | Ein Schrittstatusoperand darf nur einmal pro Programm mit einer STL-Anweisung programmiert werden.
- | Die STL-Anweisung darf nicht in einem Interrupt-Programm eingesetzt werden.
- | Verwenden Sie keine Sprunganweisungen innerhalb eines Schrittstatus.
- | Jede Schrittsteuerung muß mit der RET-Anweisung beendet werden.
- | Der zuletzt aktivierte Schrittstatusoperand sollte mit einer RST-Anweisung zurückgesetzt werden, oder es muß eine Weiterschaltung zurück an den Anfang der Schrittfolge erfolgen.

Zulässige Anweisungen innerhalb eines Schrittstatus

In der folgenden Tabelle sind die Anweisungen des Grundbefehlssatzes aufgeführt, die zwischen STL-Anweisungen bzw. zwischen einer STL- und einer RET-Anweisung eingesetzt werden dürfen.

Status		Anweisungen		
		LD, LDI, OUT, NOP, AND, ANI, SET, RST, OR, ORI, PLS, PLF	ANB, ORB, MPS, MRD, MPP	MC, MCR
Initialisierungsstatus		zulässig	zulässig	nicht zulässig
Programmverzweigung	Ausgänge	zulässig	zulässig	nicht zulässig
	Weiterschaltbedingung	zulässig	nicht zulässig	nicht zulässig

Tab. 5-2: Zulässige Anweisungen innerhalb eines Schrittstatus

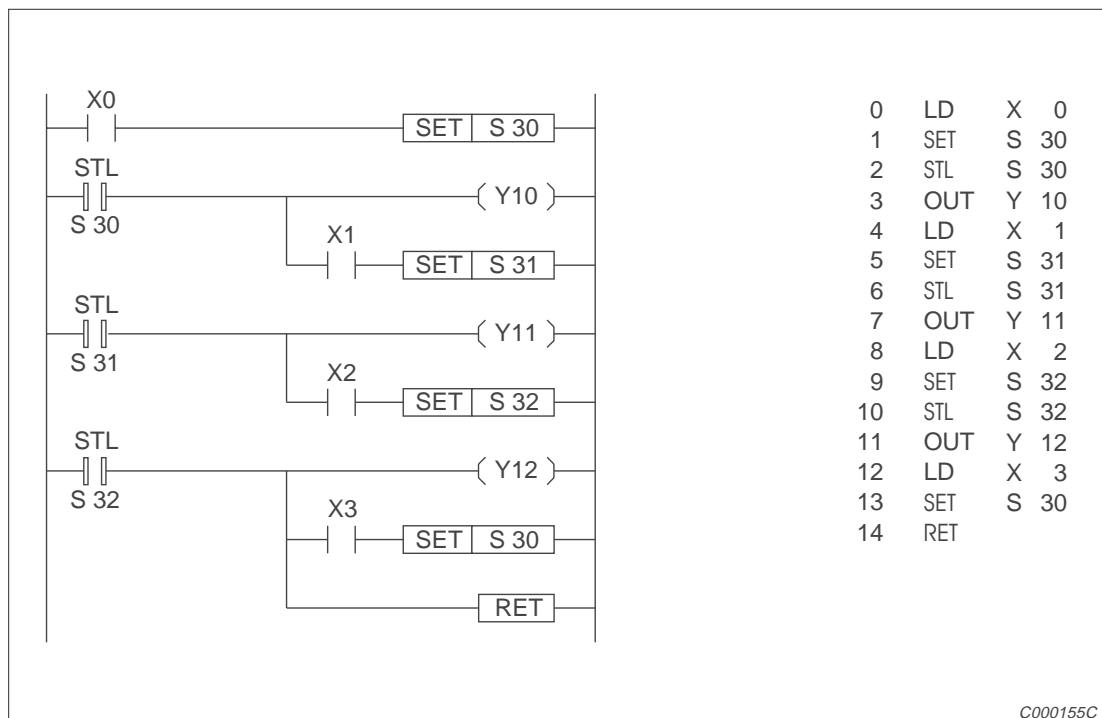


Abb. 5-5: Programmierbeispiel zum Einsatz der STL-, RET-Anweisungen

Ausgänge mehrfach belegen

Der gleiche Ausgang kann mit verschiedenen STL-Anweisungen bzw. Schrittstatusoperanden angesprochen werden.

Beispiel ▾

Ausgänge mehrfach belegen

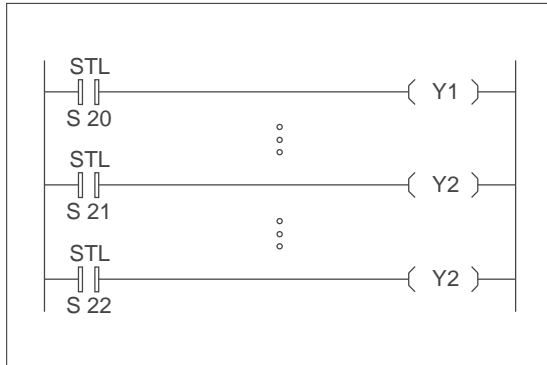


Abb. 5-6:

Ausgänge mehrfach belegen

C000138C

Im obenstehenden Programmausschnitt wird der gleiche Ausgang (Y2) über verschiedene STL-Anweisungen bzw. Schrittstatusoperanden (S21 und S22) angesprochen.

Y2 wird eingeschaltet, wenn S21 oder S22 aktiv ist. Y2 wird ausgeschaltet, wenn S21 und S22 nicht aktiv sind. Die Doppelbelegung ist in diesem Fall unproblematisch, da die Schritte 21 und 22 nicht gleichzeitig aktiv sein können. △

Rücksetzfunktion der Weiterschaltbedingung

Sobald der Status S durch die STL-Anweisung gesetzt ist, wird die Weiterschaltbedingung des vorangegangenen Status zurückgesetzt. Das bedeutet, daß in einem Programmzyklus der aktuelle wie auch der nachfolgende Status für eine sehr kurze Zeit gleichzeitig gesetzt sein können.

HINWEIS

Wenn die aufeinanderfolgenden Operanden nicht gleichzeitig aktiv sein dürfen, empfiehlt es sich, die Operanden durch eine Verriegelung zu schützen.

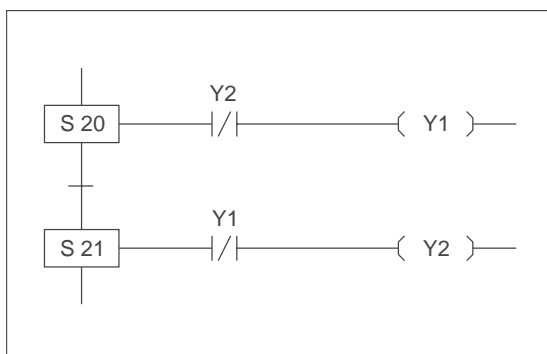


Abb. 5-7:

Verriegelungsmaßnahme, um einen gleichzeitigen Einschaltzustand zu verhindern

C000139C

Timer mehrfach belegen

In einem Programm kann ein Timer durch Einsatz von Schrittstatusoperanden mehrfach belegt werden. Ein Timer darf jedoch nicht in zwei aufeinanderfolgenden Schritten eingesetzt werden.

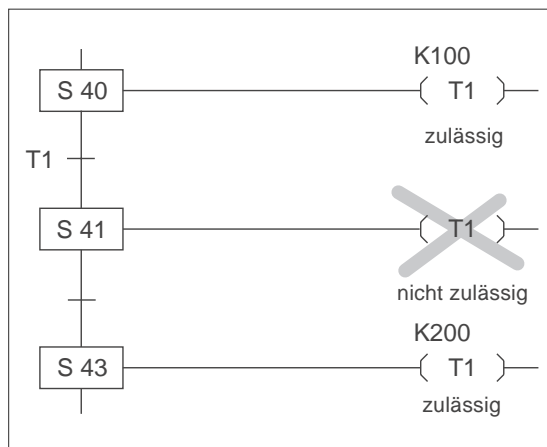


Abb. 5-8:
Timer mehrfach belegen

C000040C

Weiterschaltbedingung durch ein Impulssignal

Aufeinanderfolgende Schritte können über dieselbe Weiterschaltbedingung aktiviert werden. Hierzu ist der Einsatz einer Impulsanweisung (PLS-Anweisung) erforderlich.

Beispiel ▾

Weiterschaltbedingung durch ein Impulssignal

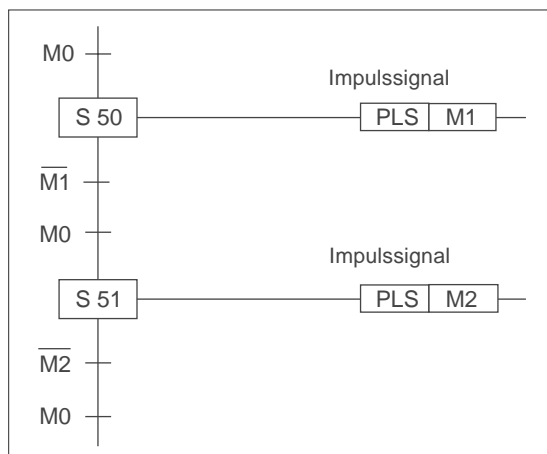


Abb. 5-9:
Weiterschaltbedingung durch ein
Impulssignal

C000041C

Das erste M0-Signal aktiviert den Schrittstatus S50 und schaltet M1 ein. M1 verhindert das unmittelbare Aktivieren des nächsten Schrittstatus. S51 wird erst aktiv, wenn das nächste M0-Signal ansteht. △

Weiterschaltbedingung durch gepulste Anweisungen (FX2N)

Bei der Verwendung der FX2N-Serie können die Weiterschaltbedingungen durch die gepulsten Anweisungen (LDP, LDF, ANP etc.) und die gepulsten Merker M2800 bis M3071 realisiert werden.

Beispiel ▾

Weiterschaltbedingung durch Verwendung des gepulsten Merkers M2800

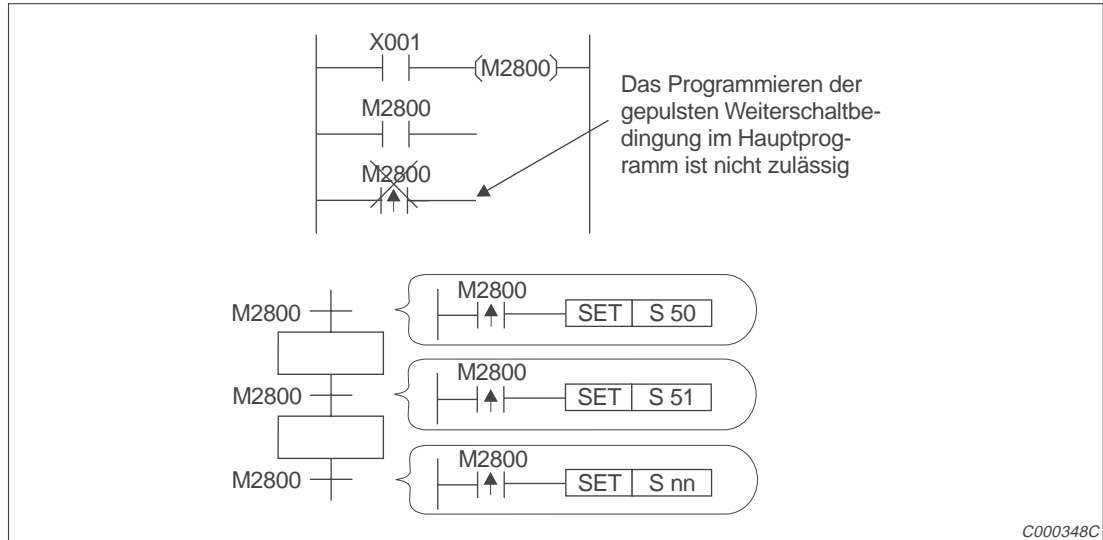


Abb. 5-10: Weiterschaltbedingung durch gepulsten Merker M2800

Durch Setzen des Merkers M2800 mit X001 wird der Schritt S51 aktiviert. Eine gleichzeitige Aktivierung des Schrittes Snn ist nicht möglich, da M2800 (gepulst) zum 2. Mal programmiert wurde. Beim nächsten Setzen von M2800 durch X001 wird der Schritt Snn aktiviert, da der Schritt S50 inaktiv ist, und der gepulste Merker M2800 somit nur einmal aktiv vorhanden ist.

△

5.3 Schrittstatus initialisieren

Jeder Schrittstatus erfordert eine Initialisierung. Dafür stehen beispielsweise die Initialisierungsoperanden S0 bis S9 zur Verfügung. Mit Hilfe der Initialisierungsoperanden können verschiedene Schrittabläufe innerhalb eines STL-Programms erstellt werden, um z. B. unterschiedliche Betriebsabläufe (Hand-/Automatikbetrieb, Nullpunktfahrt usw.) zu realisieren.

Beispiel ▾

Schrittstatus initialisieren

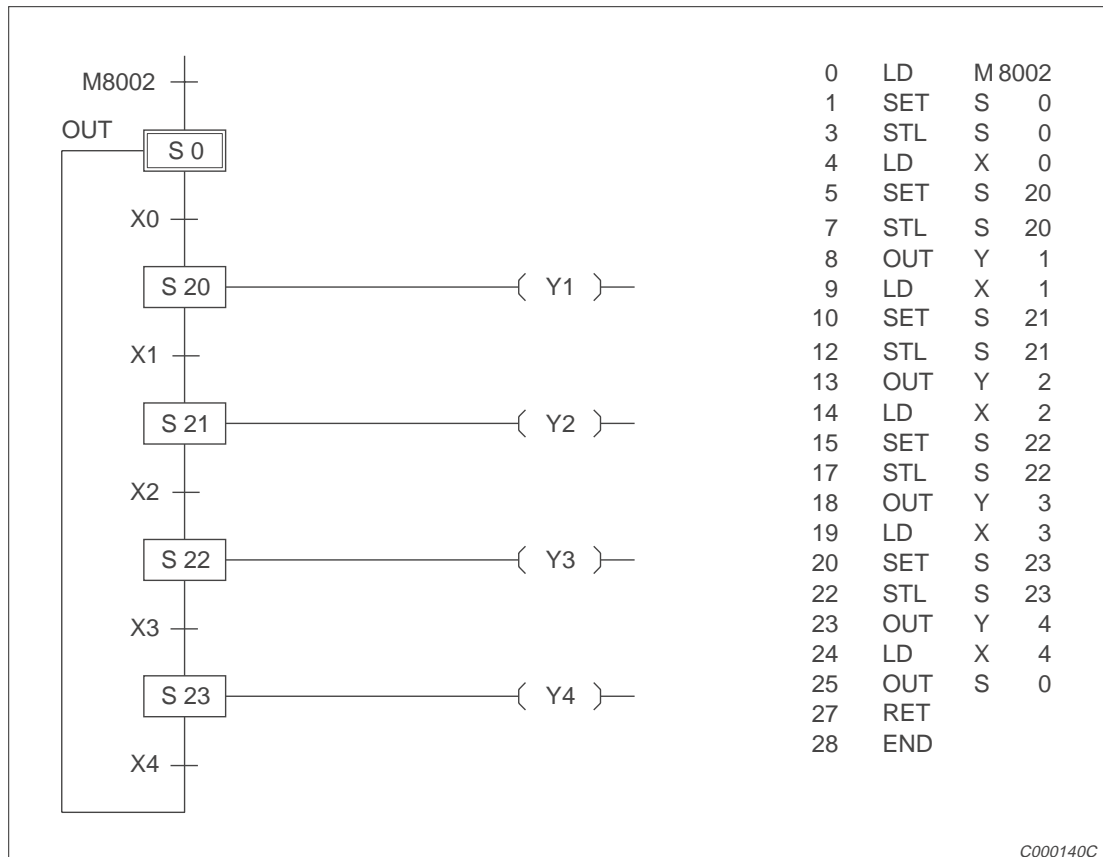


Abb. 5-11: Programmierbeispiel zum Initialisieren eines Schrittstatus

Der Merker M8002 bewirkt beim Einschalten der SPS einen definierten Systemzustand (siehe auch Abs. 10.1.1). Es findet eine Initialisierung der Schrittkette statt, indem S0 gesetzt wird.

Die Schrittbedingungen für jeden weiteren Schrittstatus werden in bereits beschriebener Art und Weise ausgeführt.

Um einen Neustart oder eine Wiederholung der Schrittkette zu bewirken, muß zunächst wieder S0 eingeschaltet werden. △

5.4 STL-Verzweigungen

Die speicherprogrammierbaren Steuerungen der FX-Familie können verschiedene, voneinander unabhängige Statusverläufe und Verzweigungen verarbeiten. Es wird unterschieden zwischen:

- Einfachverlauf
- selektive Verzweigung
- parallele Verzweigung
- Sprungverzweigung

5.4.1 Einfachverlauf

Beim Einfachverlauf wird der Schrittstatusverlauf seriell (nacheinander) abgearbeitet. Die Reihenfolge der Abarbeitung richtet sich nur nach der Stellung des Schrittstatus im Einfachverlauf und ist dadurch unabhängig von der Schrittstatusadresse.

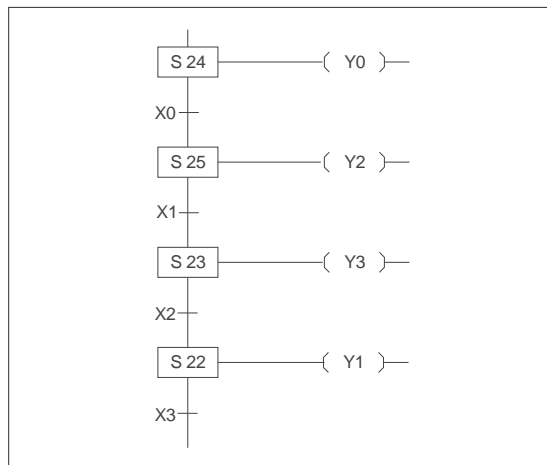
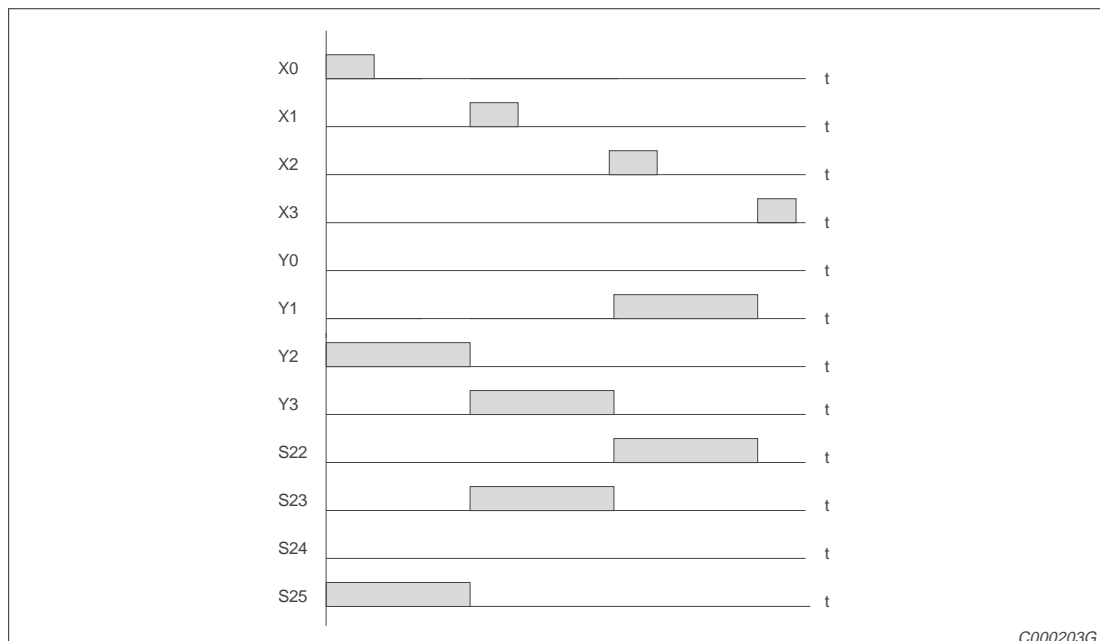


Abb. 5-12:
Beispiel für einen Einfachverlauf

C000143G



C000203G

Abb. 5-13: Zeitdiagramm des Einfachverlaufs

5.4.2 Selektive Verzweigung

Bei der selektiven Verzweigung besteht die Möglichkeit, eine **Auswahl** zwischen zwei oder mehreren Statusverläufen während einer Operation zu treffen.

Aus einem Schrittstatus heraus findet eine Verzweigung in mehrere (maximal 8) Statusverläufe statt.

In Abhängigkeit von der jeweils gesetzten Eingangsbedingung erfolgt die Auswahl, welcher Statusverlauf im Programm aktiviert werden soll. Es kann nur jeweils ein Pfad aktiv sein.

HINWEIS

Es können maximal 8 Verzweigungen ausgehend von einem Schrittoperanden programmiert werden. Die Gesamtzahl aller selektiven Verzweigungen darf 16 nicht überschreiten.

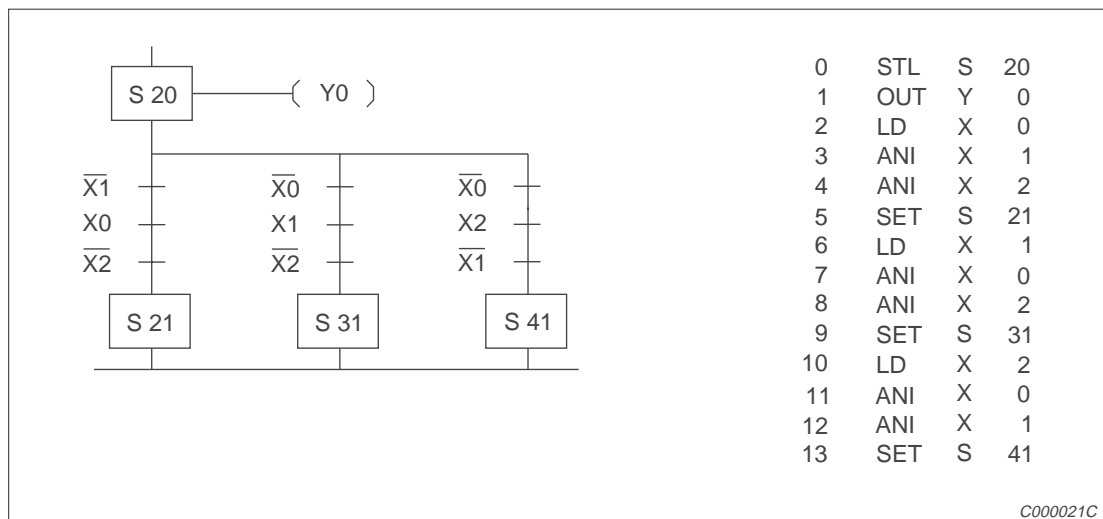


Abb. 5-14: Start einer selektiven Verzweigung

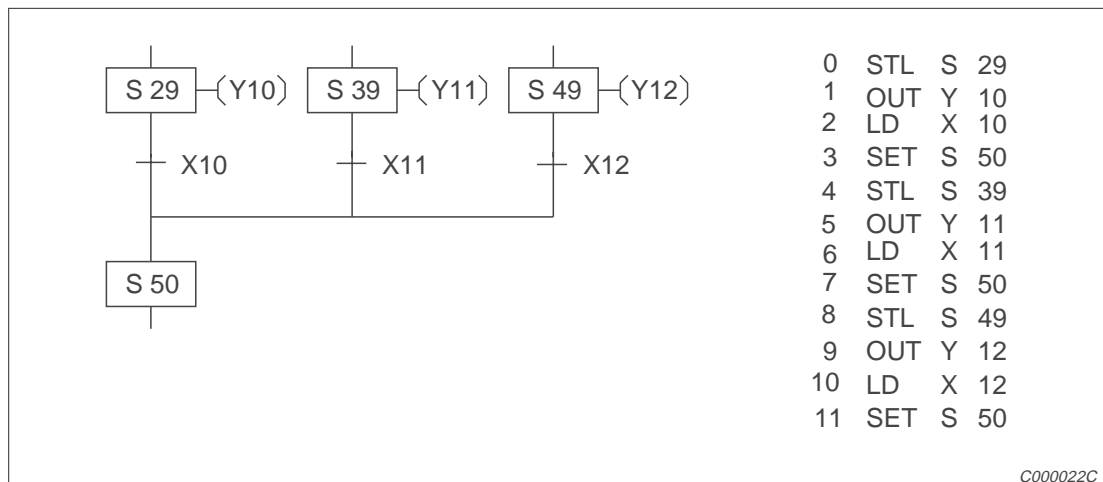


Abb. 5-15: Zusammenführen einer selektiven Verzweigung

Beispiel ▾ Flußdiagramm, Kontaktplan und Anweisungsliste einer selektiven Verzweigung.

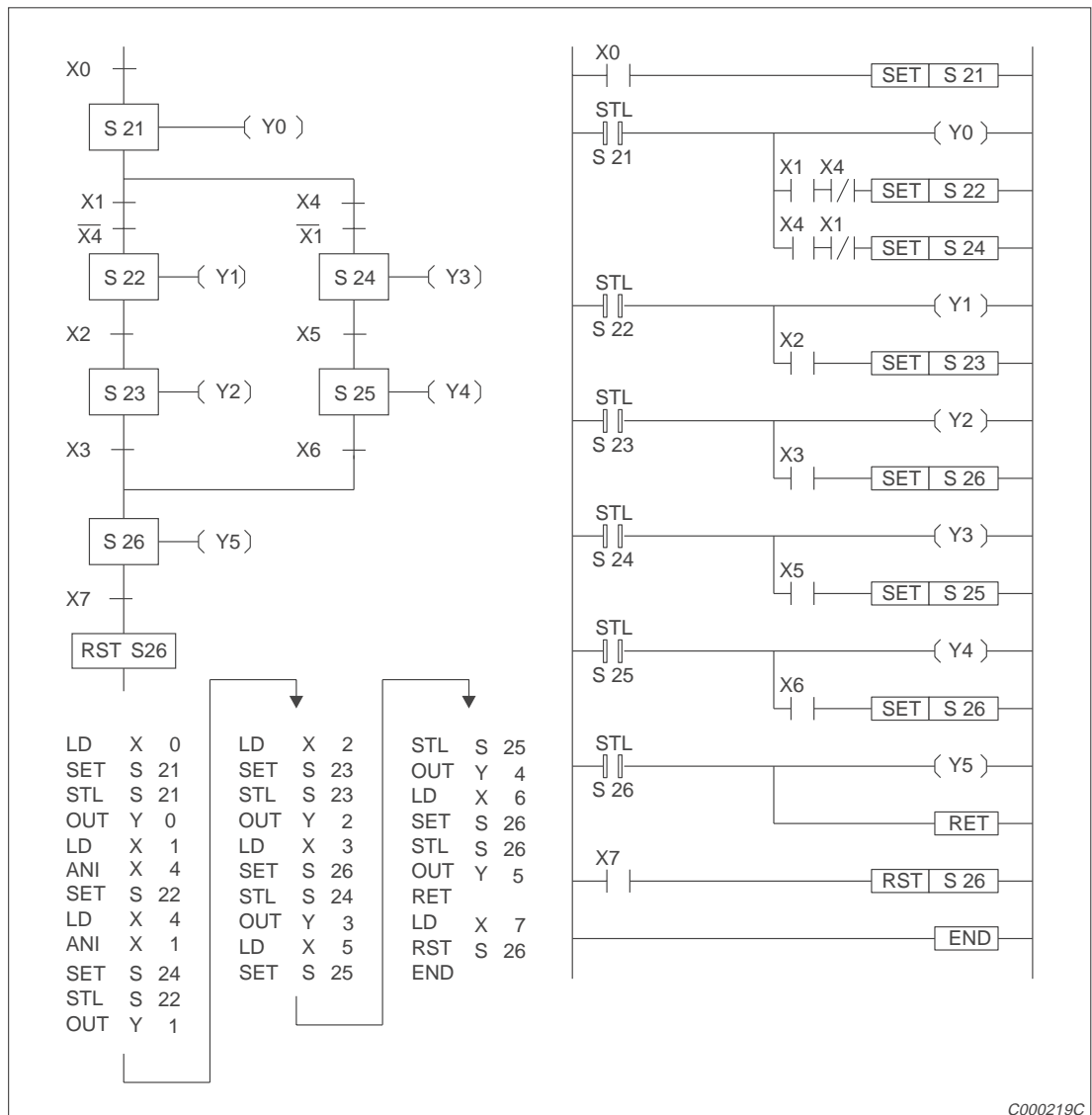


Abb. 5-16: Programmierbeispiel für eine selektive Verzweigung

Es darf hierbei immer nur eine der Funktionen ausgeführt werden. Dies ist sichergestellt, indem S21 automatisch zurückgesetzt wird, wenn entweder S22 oder S24 gesetzt ist.

S26 wird über Schritt S23 und S25 gesetzt. Dementsprechend wird beim Setzen von S26 entweder S23 oder S25 zurückgesetzt. △

5.4.3 Parallele Verzweigung

Bei der parallelen Verzweigung werden zwei oder mehr Statusverläufe **gleichzeitig abgearbeitet**. Aus einem Status heraus findet eine Verzweigung in mehrere (maximal 8) Statusverläufe statt.

HINWEIS | Die Anzahl aller Verzweigungen darf 16 nicht überschreiten.

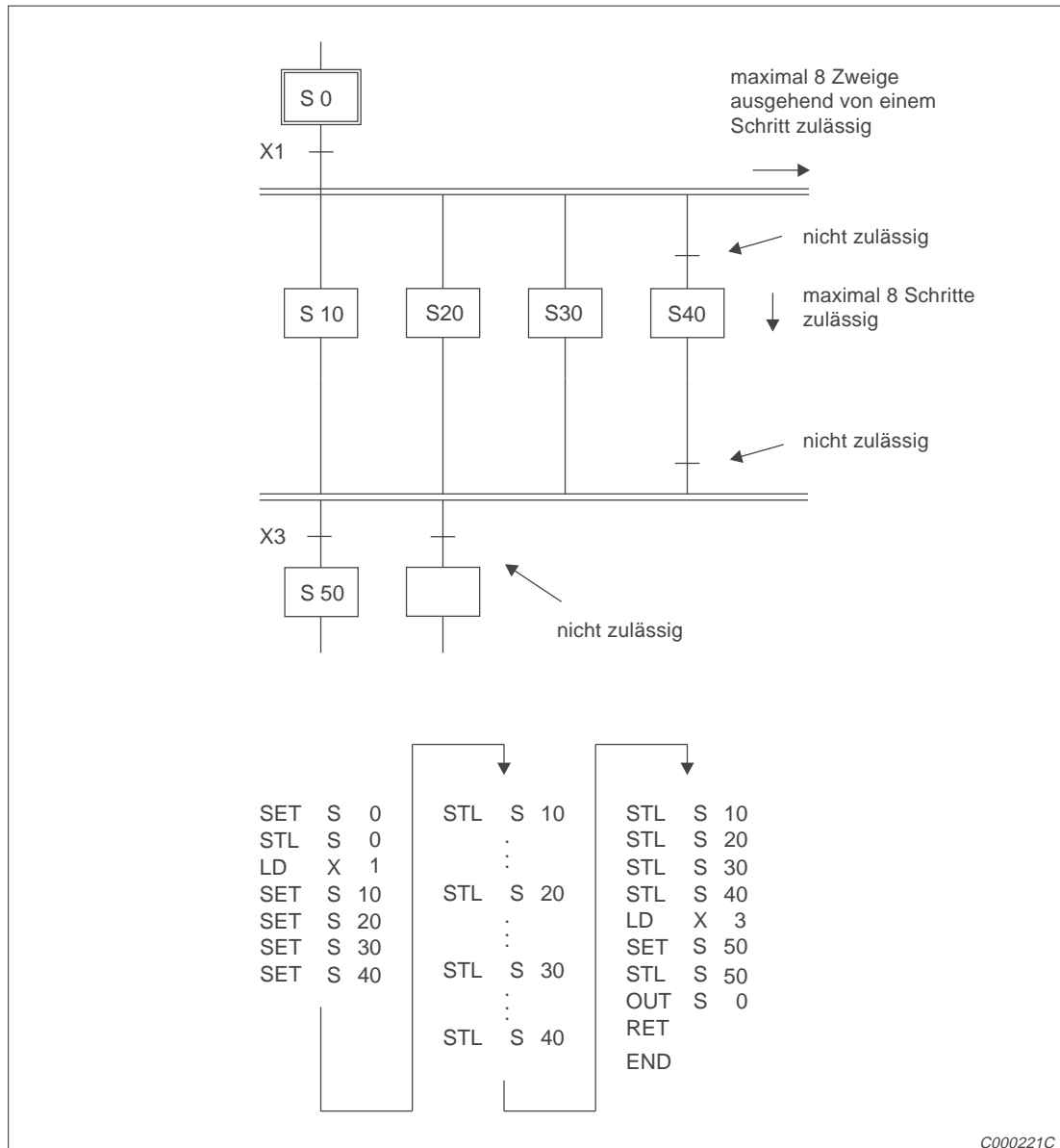


Abb. 5-17: Zulässige parallele Verzweigungen

In Abhängigkeit von der jeweils gesetzten Eingangsbedingung erfolgt die Verzweigung in die einzelnen Pfade. Im Gegensatz zur selektiven Verzweigung können bei der parallelen Verzweigung mehrere Statusverläufe gleichzeitig abgearbeitet werden.

Die geschalteten Operanden der parallelen Schritte werden erst zurückgesetzt, wenn die nach der Zusammenführung liegenden Schritte abgearbeitet werden.

HINWEISE

Nach der Verzweigung und vor der Zusammenführung sind keine Verknüpfungen zulässig.

Eine parallele Verzweigung darf maximal 8 parallele Zweige enthalten, von denen jeder Zweig aus maximal 8 aufeinanderfolgenden Schritten bestehen darf.

Innerhalb einer parallelen Verzweigung darf keine weitere selektive Verzweigung programmiert werden.

Beispiel ▾

Flußdiagramm, Kontaktplan und Anweisungsliste einer parallelen Verzweigung

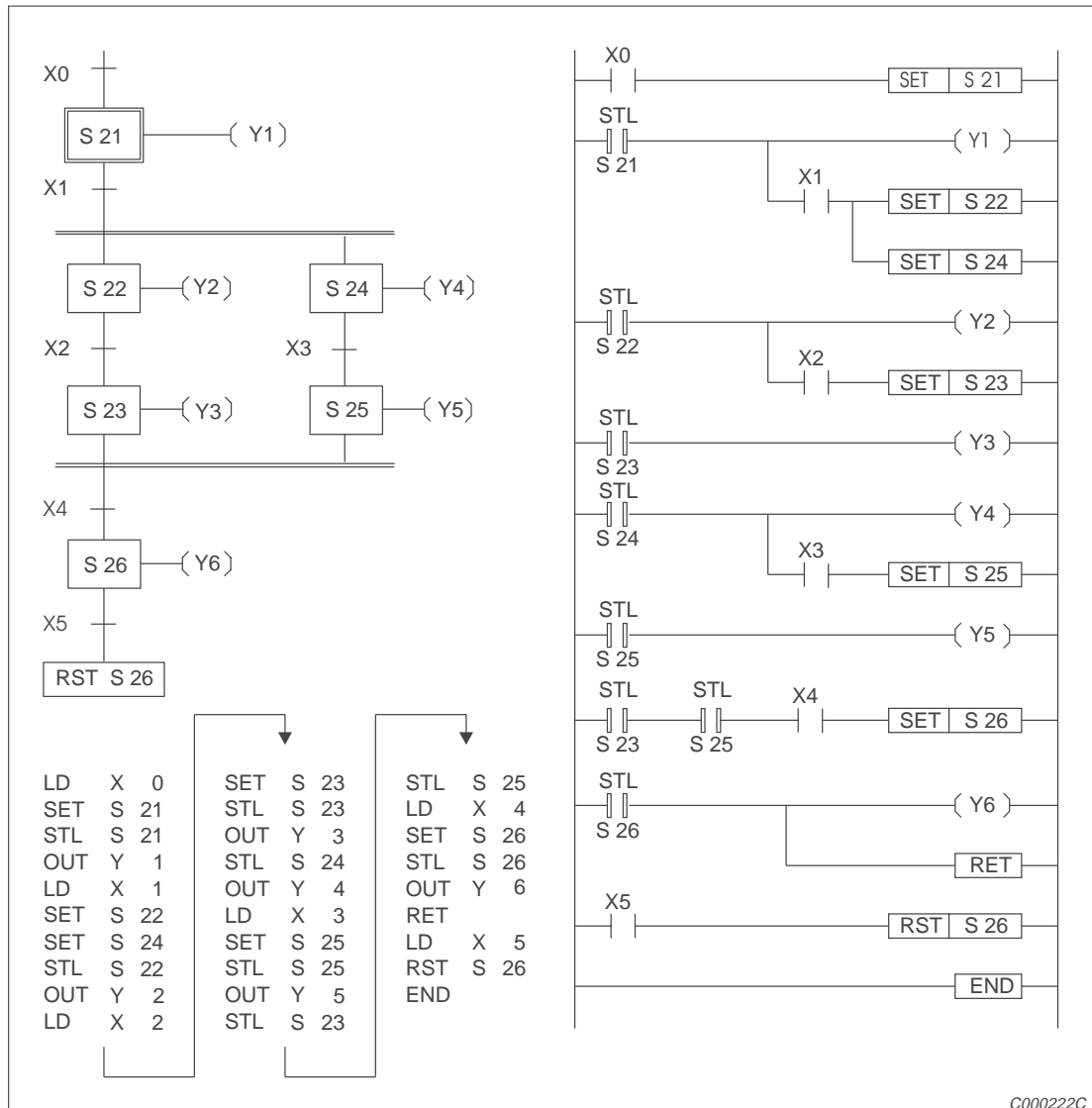


Abb. 5-18: Parallele Verzweigung

Der Schritt S26 ist in Abhängigkeit von X4 erst nach Ausführung der Schritte S23 und S25 durchführbar. △

5.4.4 Kombination aus selektiver und paralleler Verzweigung

Selektive und parallele Verzweigungen können in einem STL-Programm kombiniert werden.

Beispiel ▾

Kombination aus selektiver und paralleler Verzweigung

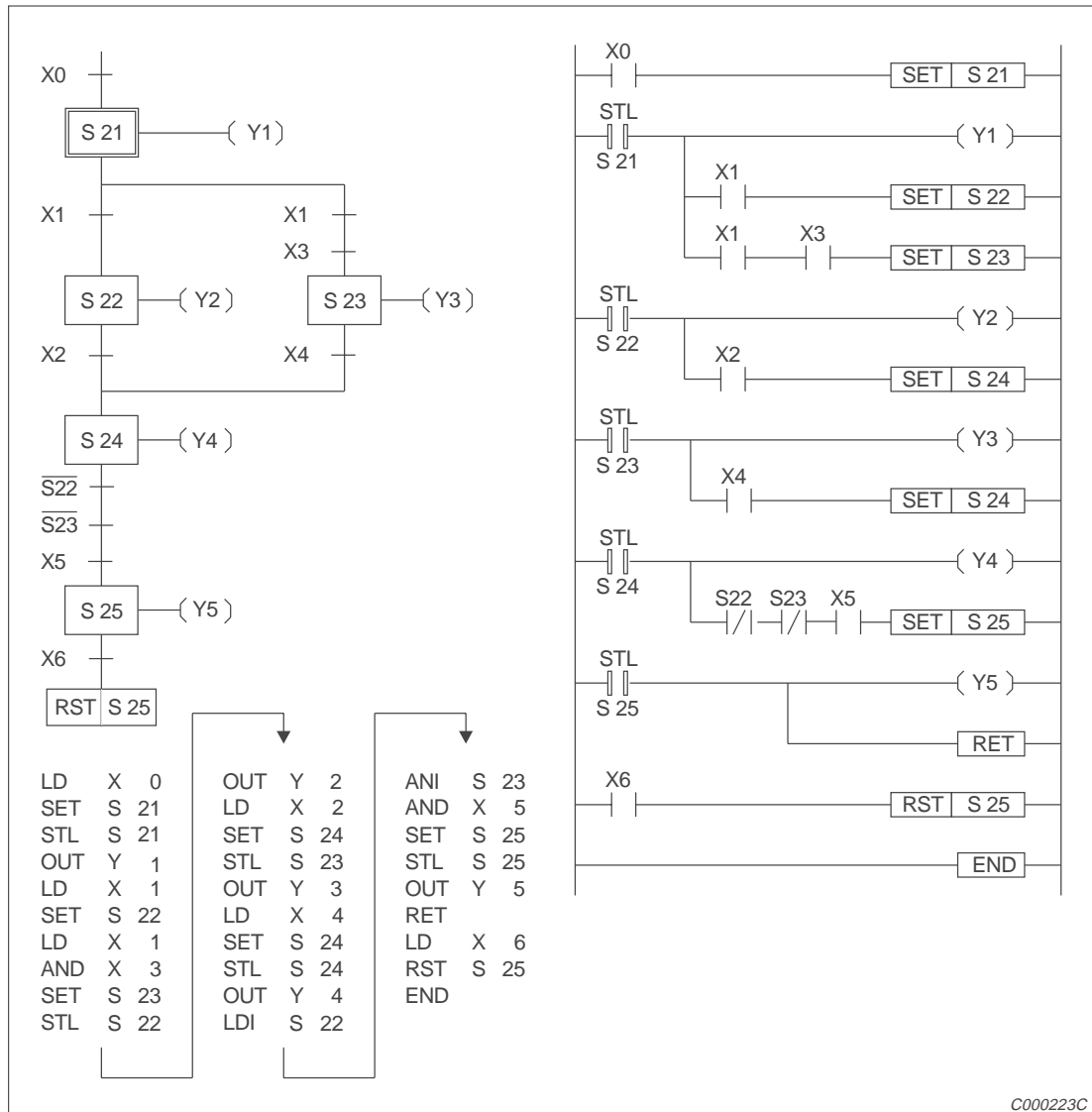


Abb. 5-19: Kombination aus selektiver und paralleler Verzweigung

Wird im Beispiel X3 gesetzt, ist die Bedingung für eine Parallelverzweigung erfüllt. Ist X3 nicht gesetzt, erfolgt eine selektive Programmbearbeitung, d.h., S24 kann nur über S22 gesetzt werden.

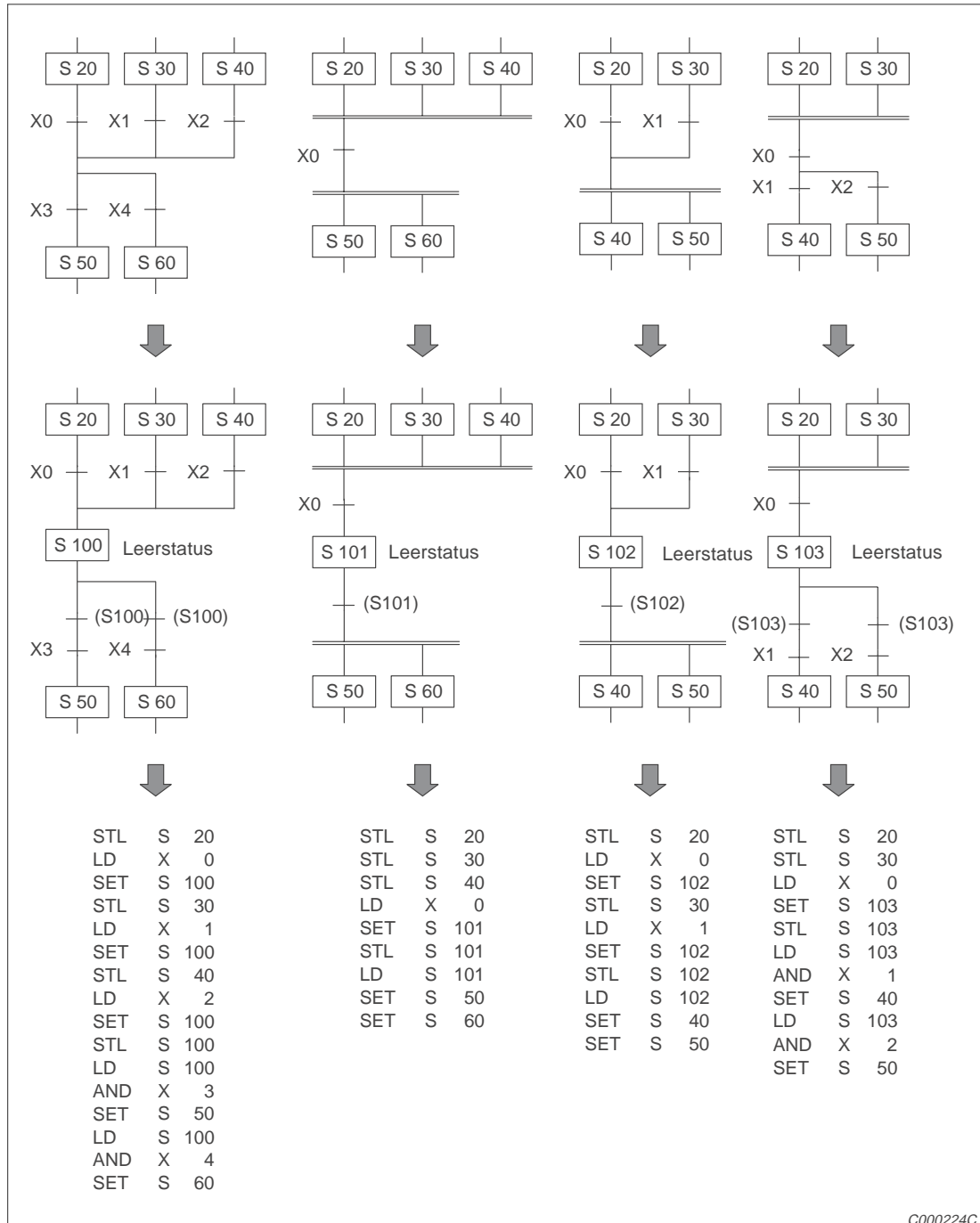
S24 wird nur gesetzt, wenn S22 oder S23 zurückgesetzt sind.

S25 wird nur gesetzt, wenn S22 und S23 zurückgesetzt sind.



5.4.5 Leerstatus programmieren

Für die Realisierung einiger Schrittabläufe ist die Programmierung eines Leerstatus notwendig. Diese Möglichkeit trägt zu einer besseren Übersicht des Programmablaufs sowie zu einer Einsparung von Programmschritten bei.



C000224C

Abb. 5-20: Leerstatus programmieren

5.4.6 Sprungverzweigung

Es besteht die Möglichkeit, Teilbereiche einer Statusfolge zu überspringen oder eine Programmschleife mehrfach zu durchlaufen.

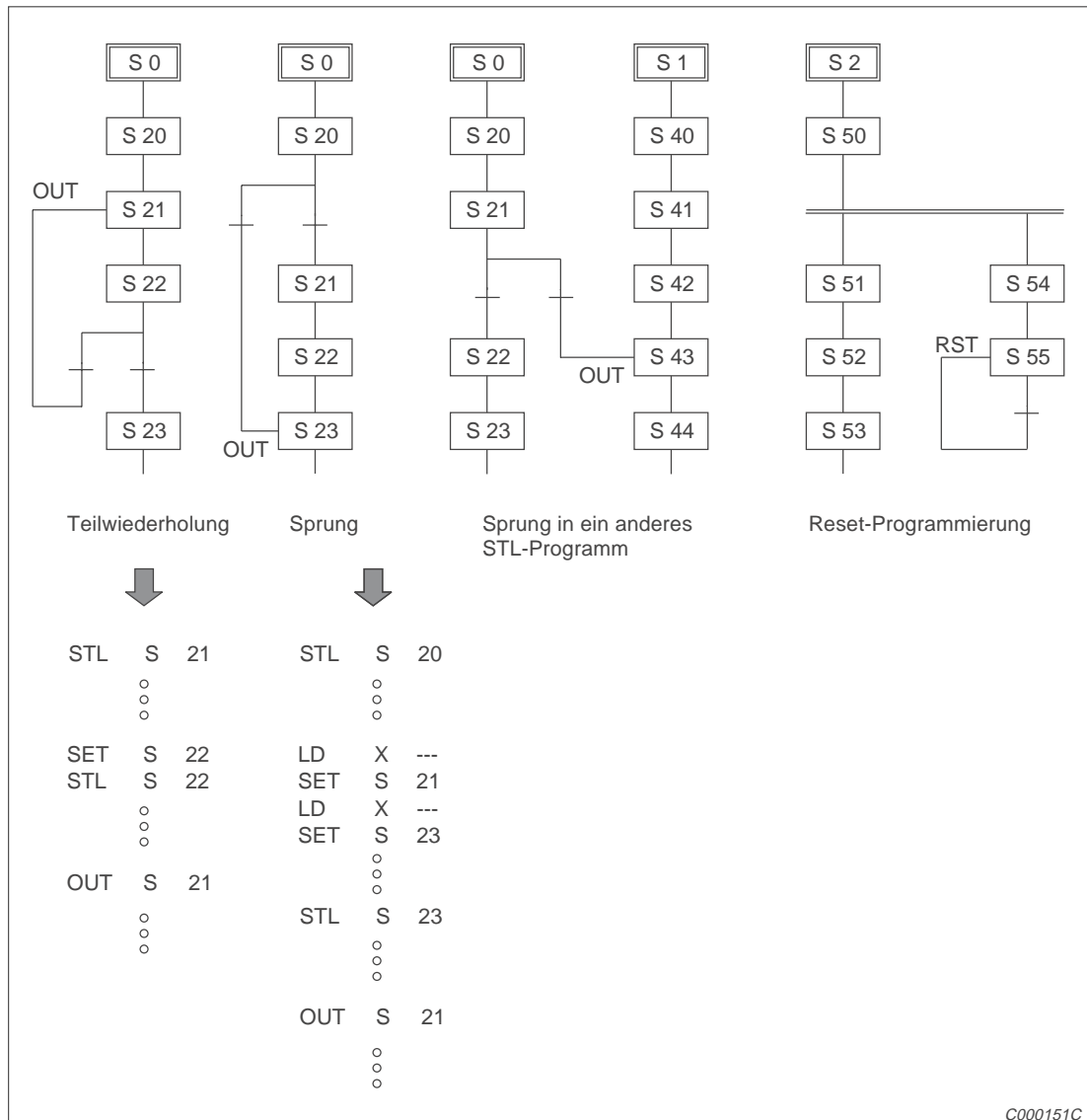


Abb. 5-21: Programmierbeispiele verschiedener Möglichkeiten einer Sprungverzweigung

Weiterschaltung in eine andere Schrittkette

Bei der Weiterschaltung von einer Schrittkette in eine andere Schrittkette kann anstatt einer SET-Anweisung auch eine OUT-Anweisung programmiert werden (siehe OUT S31 im Beispiel zur Schrittkette I). Diese Alternative hat keine Auswirkung auf die interne Programmabarbeitung der Steuerung.

Beispiel ▾

Weiterschaltung in eine andere Schrittkette

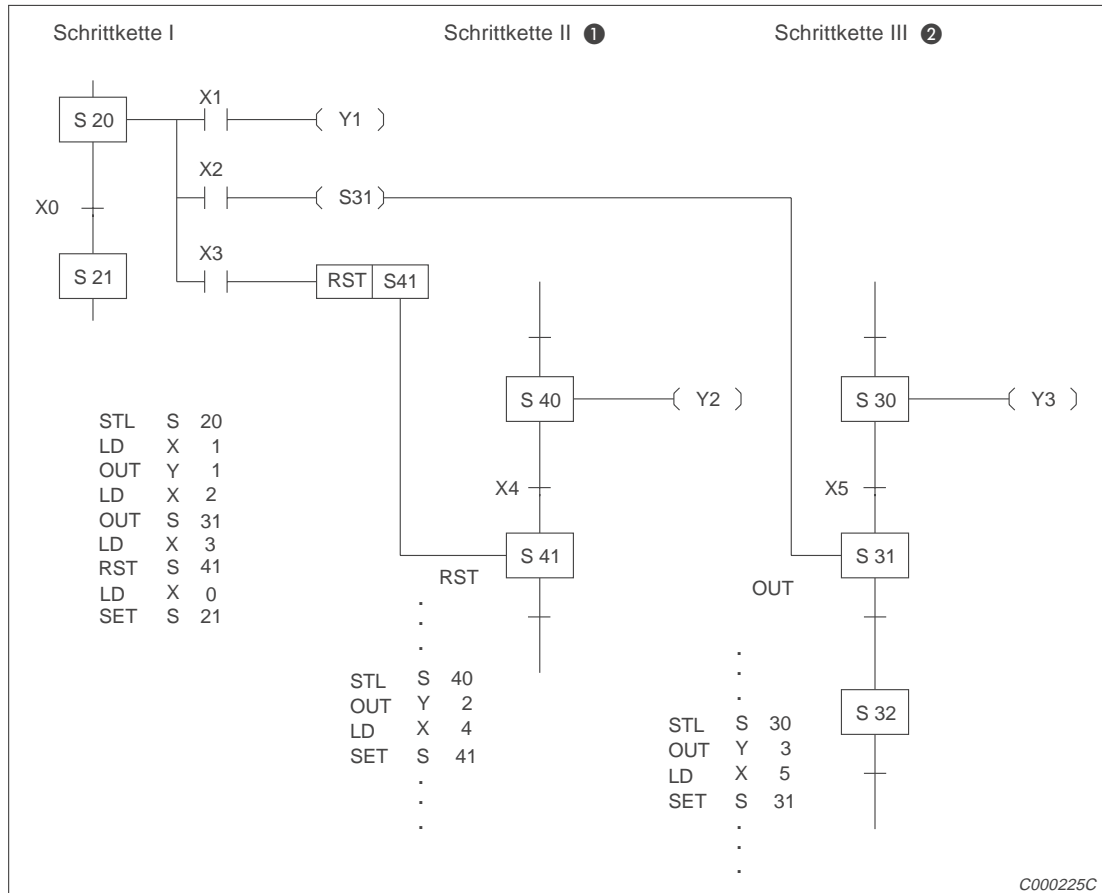


Abb. 5-22: Weiterschaltung in eine andere Schrittkette

- ① Der Schrittstatus S41 in der Schrittkette II wird über S40 und den Eingang X4 gesetzt. Der Schrittstatus S41 wird zurückgesetzt, wenn S20 und der Eingang X3 in der Schrittkette I eingeschaltet sind.
 Ist der Rücksetzvorgang abgeschlossen, befindet sich die Schrittkette weiterhin im Schrittstatus S20, welcher von S41 nicht beeinflusst wird.
- ② Der Schrittstatus S31 in der Schrittkette III wird gesetzt, wenn S20 und der Eingang X2 in der Schrittkette I gesetzt sind. S31 wird zurückgesetzt, nachdem zu S32 weitergeschaltet wurde. Der Schrittstatus S20 wird zurückgesetzt, wenn zum Schrittstatus S31 weitergeschaltet wird.



5.5 Beispiel für eine Be- und Entladekontrolle

Beispiel ▾

In diesem Beispiel wird ein Containerfahrzeug für Schüttguttransport auf einem festgelegten Abschnitt verfahren und an vorgegebener Stelle be- bzw. entladen.

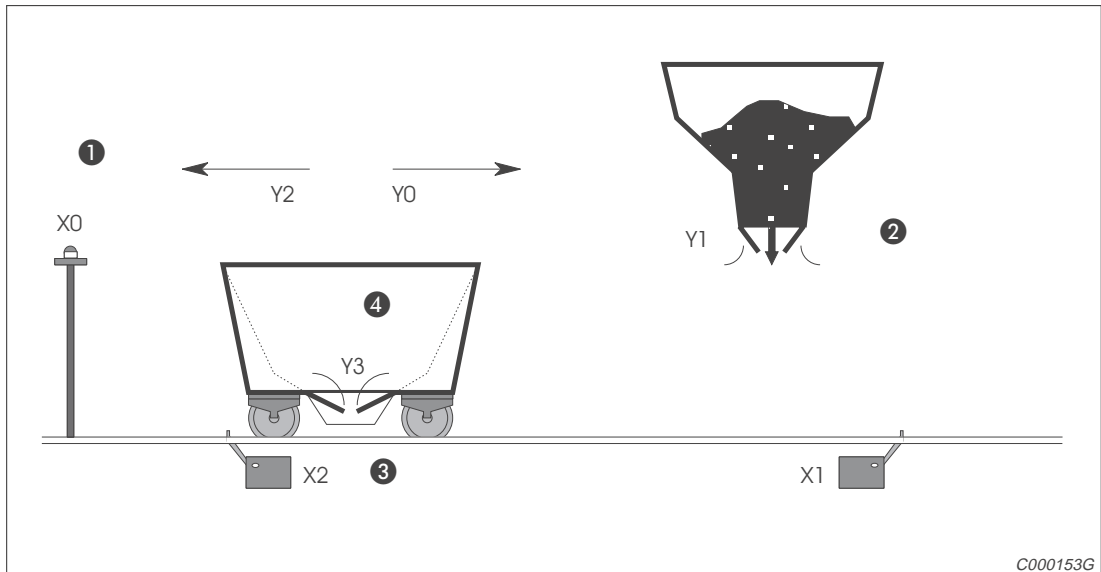


Abb. 5-23: Beispiel für eine Be- und Entladekontrolle eines Containerfahrzeuges

- ① Nach dem Betätigen der Starttaste X0 fährt der Wagen in Richtung Beladestelle und stoppt am Endschalter X1.
- ② Die Silolade öffnet sich für 7 Sekunden (Y1).
- ③ Der Wagen fährt zurück und stoppt am Endschalter X2 an der Entladestelle.
- ④ Die Entladeklappe des Wagens öffnet sich für 5 Sekunden (Y3).

Beispiel ▾

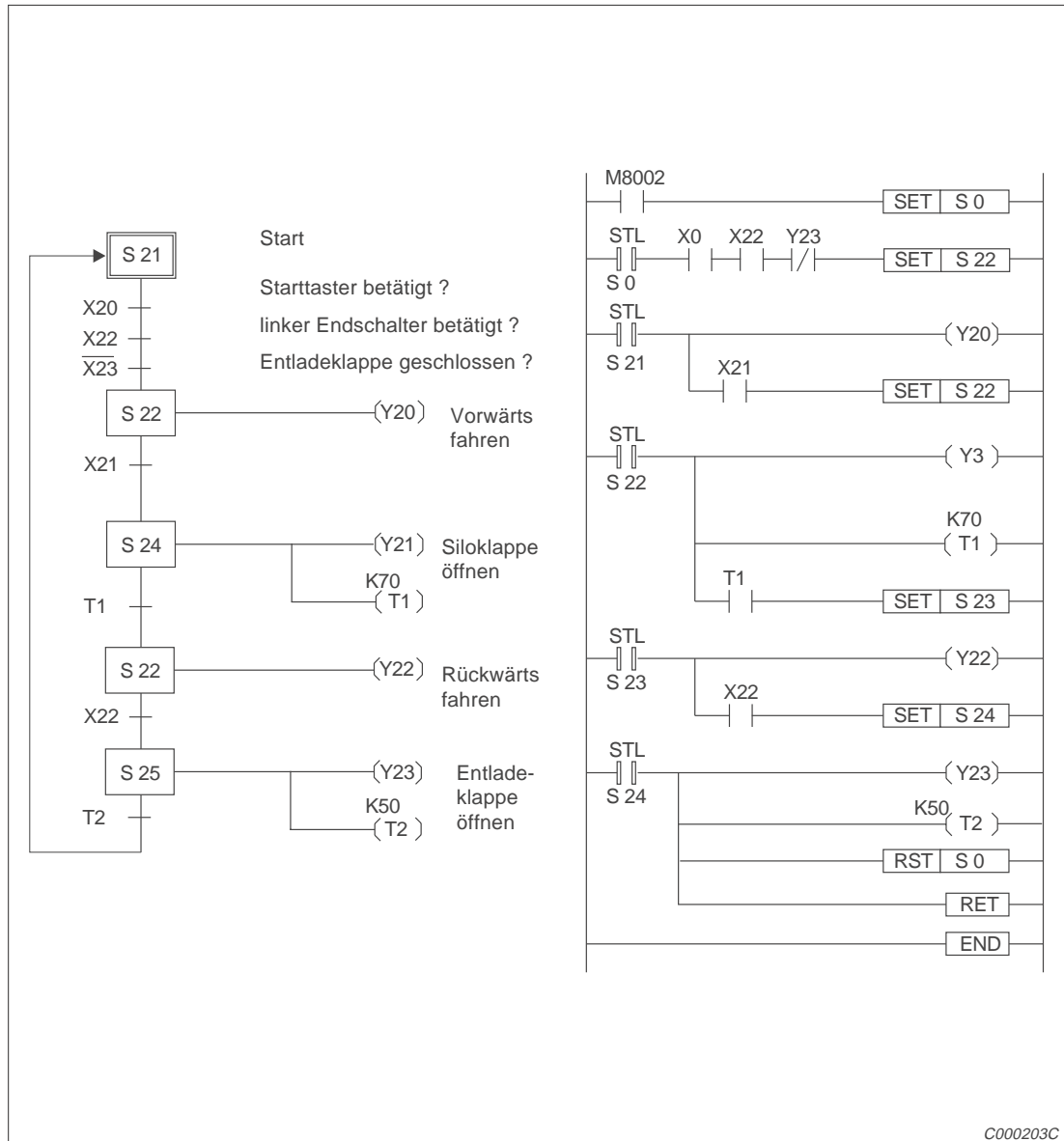


Abb. 5-24: Programmablauf der Be- und Entladekontrolle eines Containerfahrzeuges

Das obige Beispiel zeigt die Programmabfolge zum vorstehenden Beispiel.



5.6 Beispiel für einen Transportier- und Sortiervorgang

Dieses Beispiel zeigt einen Steuerungsmechanismus, bei dem unterschiedlich große Stahlkugeln aus einem Behälter gehoben und über einen Förderweg transportiert werden. Am Ende des Förderweges werden die Kugeln der Größe nach in entsprechende Behälter sortiert.

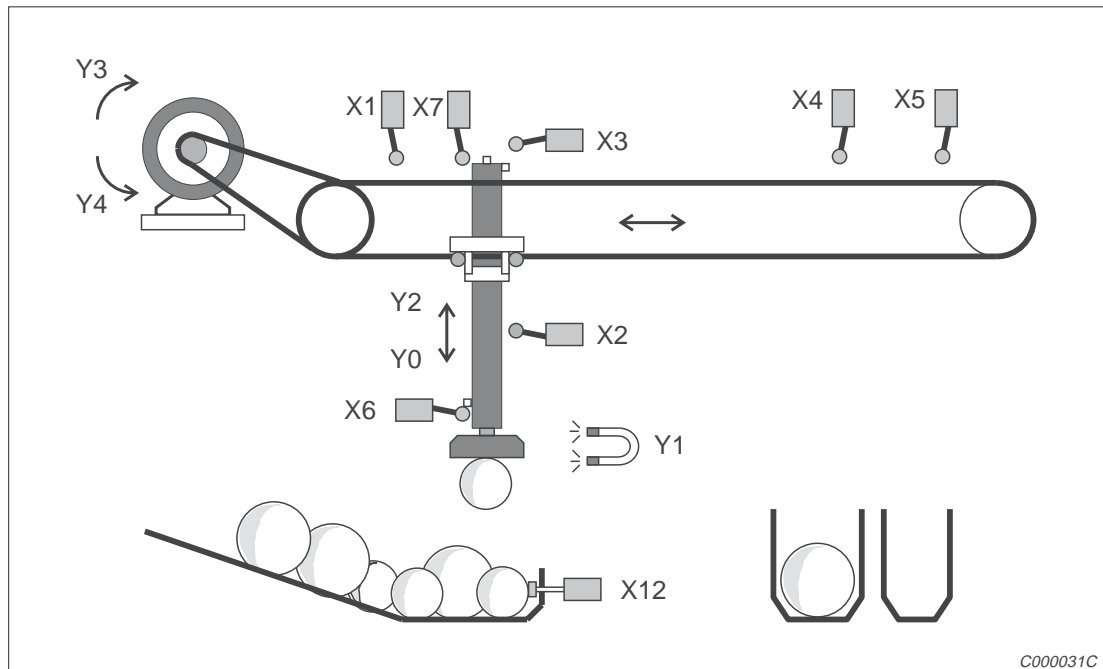


Abb. 5-25: Kugelsortiervorrichtung

- ① Der Hydraulikarm wird aus der Ausgangsposition abgesenkt (Y0=ein).
- ② Wird die untere Grenze nicht erreicht, liegt im Aufnahmeschacht eine große Kugel (X2=aus; X6=ein). Der Kontakt X2 ist geschlossen, wenn im Aufnahmeschacht eine kleine Kugel liegt.
- ③ Der Elektromagnet wird eingeschaltet (Y1=ein), und die Kugel wird aufgenommen.
- ④ Der Hydraulikarm wird angehoben (Y2=ein). Der Arm stoppt bei Erreichen der oberen Grenze (X3).
- ⑤ Der Hydraulikarm fährt nach rechts (Y3=ein).
- ⑥ Wurde eine kleine Kugel aufgenommen, stoppt der Motor bei Erreichen des Endschalters X4. Wird eine große Kugel aufgenommen, erfolgt der Motorstopp bei Erreichen des Endschalters X5.
- ⑦ Der Hydraulikarm wird abgesenkt (Y0=ein).
- ⑧ Nach Erreichen des Bodens (X6) wird der Magnet ausgeschaltet (Y1=aus).
- ⑨ Der Hydraulikarm wird bis zur oberen Grenze (X3) angehoben (Y2=ein).
- ⑩ Der Hydraulikarm wird in die Ausgangsposition gefahren (Y4=ein).
- ⑪ Die Ausgangsposition ist erreicht (X7=ein).

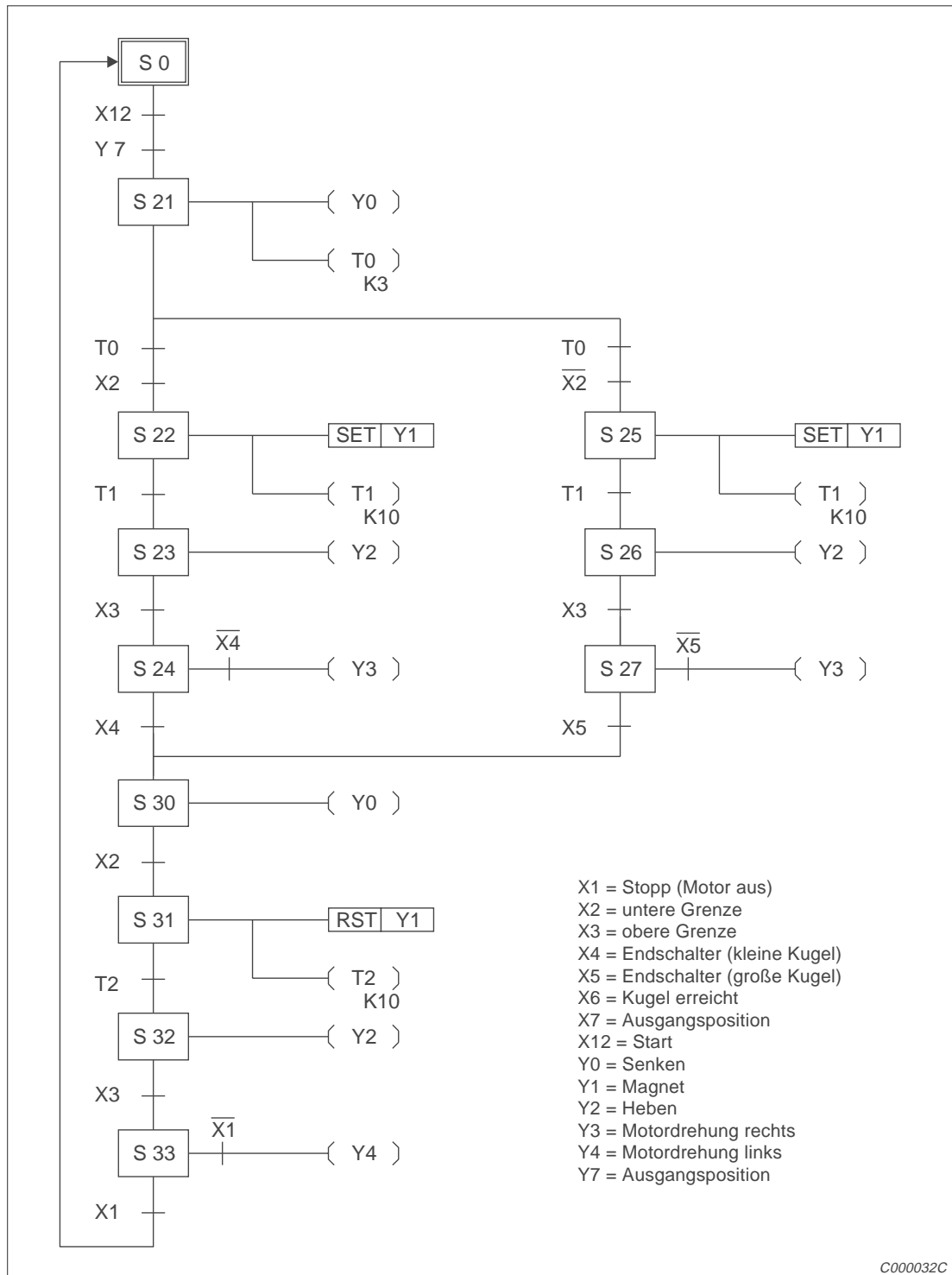


Abb. 5-26: Programmierbeispiel für die vorstehende Sortiervorrichtung

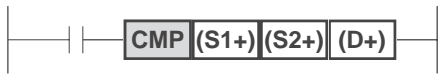
6 Applikationsanweisungen

6.1 Allgemeine Hinweise

Dieses Kapitel beschreibt die Applikationsanweisungen der FX-Familie. Mit den Applikationsanweisungen lassen sich spezielle Funktionen realisieren (z.B. Flip-Flop-Funktion, arithmetische Funktionen). Die Beschreibung einer Applikationsanweisung beginnt jeweils mit einer Übersicht in Form einer Tabelle, in der alle für die Ausführung der Applikationsanweisung wichtigen Informationen aufgeführt sind.

6.1.1 Erläuterung zur Beschreibung der Applikationsanweisungen

Dieser Abschnitt gibt eine einführende Erläuterung zur Struktur der Applikationsanweisungstabellen, die bei jeder Anweisung zu Beginn des jeweiligen Abschnitts zu finden sind.

		CMP		FNC 10			
		③ Numerische Daten vergleichen					
Operanden S1+, S2+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		D+ Y, M, S		CPU ④	FX1S	FX1N	FX2N
				●	●	●	
⑤		⑥ Puls-Anweisung (P) ●		Verarbeitung		Programmschritte	
				16 Bit	32 Bit	CMP	7
				⑦			

① Anweisung

In der oberen Zeile wird der Name der Anweisung und die zugehörige FNC-Nummer (Function number) angegeben, die bei der Programmierung in Anweisungsliste eingesetzt wird. Sie können, je nach Programmiersystem, den Anweisungsnamen oder die FNC-Nummer einsetzen.

② Kontaktplansymbol

Das Kontaktplansymbol wird bei der Kontaktplanprogrammierung verwendet. Das Kontaktplansymbol besteht aus der Anweisung und den einsetzbaren Operanden.

③ Bedeutung

Hier finden Sie eine kurze Beschreibung zur Bedeutung der Anweisung.

④ CPU

An dieser Stelle wird die MELSEC-Serie durch einen ● gekennzeichnet, mit der diese Anweisung ausführbar ist.

⑤ Operanden

Alle im Zusammenhang mit der Anweisung einsetzbaren Operanden werden in diesem Feld angegeben. Eine eingehende Beschreibung der Operanden und der Datenstruktur enthält Abs. 6.1.2 bis 6.1.4.

6 Puls-Anweisung

Befindet sich hier ein ● , kann die Ausführung der Anweisung auch bei steigender Flanke der Eingangsverknüpfung ausgeführt werden (siehe auch Abs. 6.1.5). Der Anweisung muß in diesem Fall ein „P“ nachgestellt werden.

7 Verarbeitung

Hier wird angegeben, ob es sich bei der Anweisung um eine 16-Bit- oder 32-Bit-Anweisung handelt. Bei einer 32-Bit-Anweisung wird dem eigentlichen Anweisungsnamen immer der Buchstabe „D“ vorangestellt (siehe auch Abs. 6.1.9).

8 Programmschritte

An dieser Stelle wird die Anzahl der Programmschritte angegeben, die zur vollständigen Ausführung der Anweisung erforderlich sind.

6.1.2 Beschreibung der Operanden

Bit-Operanden

Ein Bit-Operand kann zwei Signalzustände („0“ und „1“) annehmen. Sein Signalzustand kann demnach mit einem Bit (0 und 1) definiert werden.

Bit-Operanden	Operanden-kennzeichen
Eingang	X
Ausgang	Y
Merker	M
Schrittstatus	S

Tab. 6-1:
Bit-Operanden

Wortoperanden

Wortoperanden können Informationszustände annehmen, die aus mehreren Bits bestehen (numerische Datenwerte). Dabei werden 8 Bits zu einem Byte und 2 Byte zu einem Datenwort zusammengefaßt.

Wortoperanden	Operanden-kennzeichen
Timer	T
Counter	C
Datenregister	D
Indexregister	V, Z

Tab. 6-2:
Wortoperanden

6.1.3 Wortverarbeitung

Mehrere aufeinanderfolgende Bit-Operanden können in einem Datenwort zusammengefaßt werden. Dadurch besteht z. B. die Möglichkeit, die Signalzustände mehrerer Eingänge auf einmal zu verarbeiten.

Die Anzahl der Bit-Operandenadressen, die von einer Applikationsanweisung angesprochen werden sollen, wird durch die Angabe einer Konstanten K festgelegt. Bei 16-Bit-Anweisungen können bis zu 16 und bei 32-Bit-Anweisungen bis zu 32 Operandenadressen in Einheiten von je 4 Operanden vorgegeben werden. Die Anzahl der zusammengefaßten Operandenadressen wird als Blocklänge definiert.

Für 16-Bit-Anweisungen liegt die Blocklänge im Bereich von K1 bis K4.

Blocklänge	Anzahl der Adressen
K1	4
K2	8
K3	12
K4	16

Tab. 6-3:
Blocklänge bei 16-Bit-Anweisungen

Für 32-Bit-Anweisungen liegt die Blocklänge im Bereich von K1 bis K8.

Blocklänge	Anzahl der Adressen
K1	4
K2	8
K3	12
K4	16
K5	20
K6	24
K7	28
K8	32

Tab. 6-4:
Blocklänge bei 32-Bit-Anweisungen

Angabe der Startadresse eines Blocks

Die Angabe der Startadresse legt den Blockbeginn fest. Bei der Angabe der Startadresse kann jede Zahl verwendet werden.

HINWEIS

Verwenden Sie bei der Festlegung von Eingängen X und Ausgängen Y möglichst nur Startadressen, die ein Vielfaches von 10 sind (z.B. X0, X10 usw.).

Bei der Festlegung der Operanden M und S sollten Sie möglichst Startadressen angeben, die ein Vielfaches von 8 sind.

Beispiel ▾

Einteilung der Blocklängen und Startadressen.

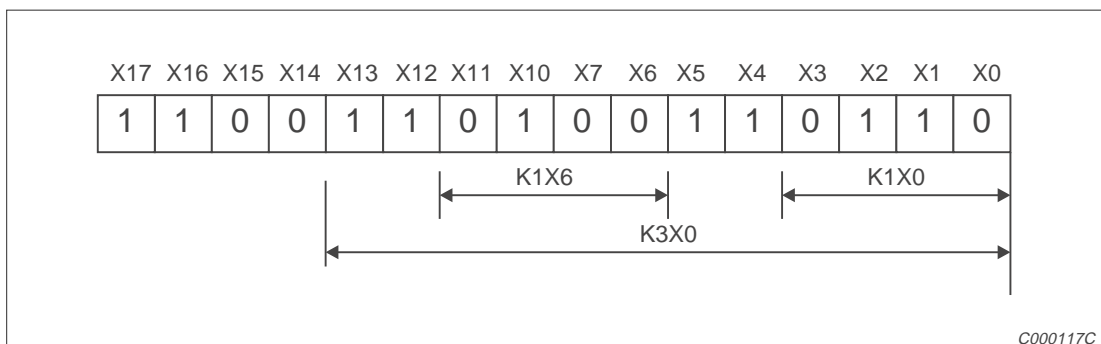


Abb. 6-1: Beispiel zur Einteilung der Blocklängen und Startadressen

K1X0: X0 bis X3 → 4 Eingänge, Startadresse X0

K1X6: X6 bis X11 → 4 Eingänge, Startadresse X6

K3X0: X0 bis X13 → 12 Eingänge, Startadresse X0



6.1.4 Datenstruktur

Quelldaten (S)

Quelldaten sind Daten, die mit einer Applikationsanweisung verarbeitet werden sollen. Die Quelldaten beinhalten eine oder mehrere Operandenadressen und können aus Konstanten und/oder Bit- oder Wortoperanden bestehen.

Konstanten sind numerische Werte, die zur Ausführung einer bestimmten Operation vorgegeben werden. Der Wert einer Konstante wird bei der Programmerstellung festgelegt und kann während der Programmverarbeitung nicht mehr geändert werden.

Mit Bit- oder Wortoperanden wird die Operandenadresse festgelegt, in der die Daten enthalten sind, die verarbeitet werden sollen. Eine Änderung der Daten ist während der Programmverarbeitung jederzeit möglich.

Datengruppe		Kennzeichen
deutsche Bezeichnung	englische Bezeichnung	
Quelldaten	Source	(S)
Quelldaten 1	Source 1	(S1)
Quelldaten 2	Source 2	(S2)

Tab. 6-5:
Kennzeichnung der Quelldaten

Zielkosten (D)

Zielkosten sind Daten, die das Operationsergebnis nach Ausführung einer Applikationsanweisung enthalten. Auch Zielkosten bestehen aus einer oder mehreren Operandenadressen und können aus Bit- oder Wortoperanden bestehen. Ein möglicher Adressenbereich der Zielkosten wird über Konstanten festgelegt und muß der Größe des Adressenbereichs der Quelldaten entsprechen.

Datengruppe		Kennzeichen
deutsche Bezeichnung	englische Bezeichnung	
Zielkosten	Destination	(D)
Zielkosten 1	Destination 1	(D1)
Zielkosten 2	Destination 2	(D2)

Tab. 6-6:
Kennzeichnung der Zielkosten

6.1.5 Ausführung von Applikationsanweisungen

Es gibt zwei Möglichkeiten, eine Anweisung auszuführen:

- Das Ausführungssignal kann z.B. als statisches Signal anstehen. Wenn das Signal eingeschaltet ist, wird die Anweisung ausgeführt.
- Das Ausführungssignal kann außerdem als ansteigende oder abfallende Flanke anstehen. Die Anweisung kommt nur dann zur Ausführung, wenn ihr Eingangssignal von „0“ nach „1“ bzw. von „1“ nach „0“ wechselt. Dies kann z. B. durch eine vorgeschaltete Impulsfunktion (PLS-, PLF-Anweisung) erreicht werden.

Beispiel ▾

Ausführung durch Merker, der für einen Zyklus gesetzt ist

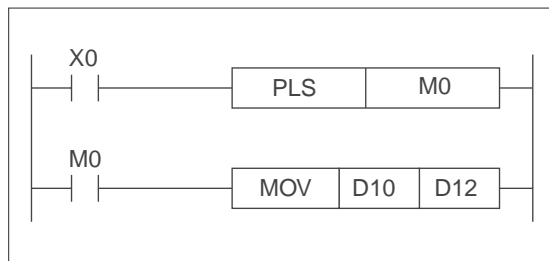


Abb. 6-2:

Ausführung durch Merker, der für einen Zyklus gesetzt ist

C000118C



Beispiel ▾

Ausführung durch Impulssignal

Zusätzlich zu der bereits gezeigten Möglichkeit verfügen die MELSEC-Steuerungen über den Befehlsparameter „P“, der die Flankenerkennung bei steigender Flanke ermöglicht.

Die Funktion des Parameters entspricht der Anweisung „PLS“ des Grundbefehlssatzes.

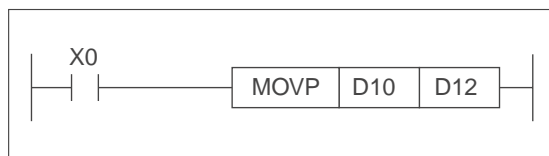


Abb. 6-3:

Ausführung durch Impulssignale

C000003C

Die Funktion dieses Beispiels ist identisch mit Abb. 6-3.

Die MOV-Anweisung wird ausgeführt, wenn am Eingang X0 ein Signalwechsel von „0“ nach „1“ stattfindet. Die Anweisung wird **nur einmal ausgeführt**. Erst bei einem weiteren Signalwechsel von „0“ nach „1“ wird die Anweisung erneut ausgeführt.

Die Anweisung wird nicht ausgeführt, wenn X0 ausgeschaltet ist.



Beispiel ▾

Ausführung durch statisches Signal

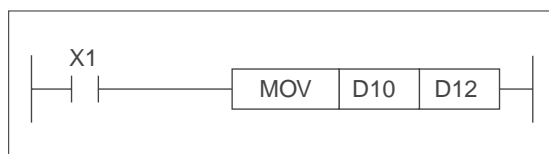


Abb. 6-4:

Ausführung durch statisches Signal

C000121C

Die MOV-Anweisung wird ausgeführt, wenn am Eingang X1 ein „1“-Signal ansteht. Die Anweisung wird solange in **jedem Programmzyklus ausgeführt**, wie das „1“-Signal ansteht.

Die Anweisung wird nicht ausgeführt, wenn X1 ausgeschaltet ist.



6.1.6 Einsatz der Indexregister V, Z

Die Indexregister V und Z werden verwendet, um bei Transfer- und Vergleichsanweisungen zur Operandenadresse einen Indexwert zu addieren.

Die Indexregister V und Z sind 16-Bit-Register.

In 32-Bit-Anweisungen müssen die beiden Indexregister V und Z kombiniert eingesetzt werden. Z speichert die unteren 16 Bit, und V speichert die oberen 16 Bit. Als Zieladresse ist das Indexregister Z anzugeben. Indexregister selbst können nicht indiziert werden.

HINWEIS

Operanden, bei denen eine Indizierung vorgenommen werden kann, sind mit einem Pluszeichen gekennzeichnet: z. B. (S+) und (D+)

Beispiel ▾

Einsatz der Indexregister V, Z

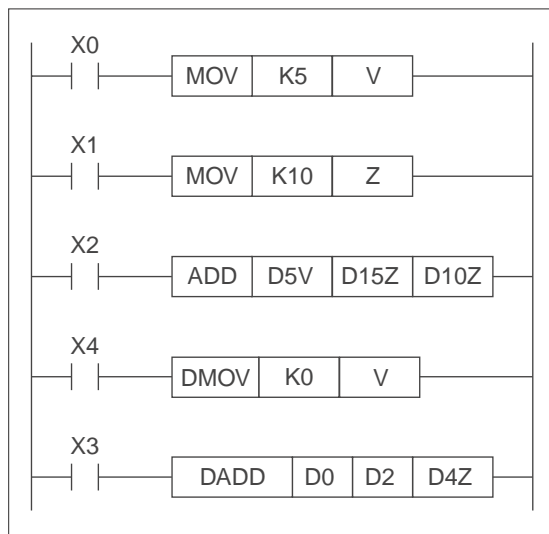


Abb. 6-5:

Programmierbeispiel zum Einsatz der Indexregister V, Z

C000120C

- Der Wert 5 (K5) wird mit der MOV-Anweisung in das Indexregister V übertragen.
- Der Wert 10 (K10) wird mit der MOV-Anweisung in das Indexregister Z übertragen.
- D5V soll zu D15Z addiert werden. Das Ergebnis soll im Datenregister D10Z abgelegt werden.
- Berechnung der Datenregister:

$$V = 5 \text{ (K5)}$$

$$Z = 10 \text{ (K10)}$$

$$D5V = D10 \text{ (D5 + V = D5 + 5 = D10)}$$

$$D15Z = D25 \text{ (D15 + Z = D15 + 10 = D25)}$$

$$D10Z = D20 \text{ (D10 + Z = D10 + 10 = D20)}$$

- Der Wert 0 (K0) wird mit der MOV-Anweisung in das Index-Register V übertragen. Es findet eine 32-Bit-Operation statt.
- D0 und D1 werden zu D2 und D3 addiert. Das Ergebnis wird in den Datenregistern D14 und D15 abgespeichert.

△

6.1.7 Bedeutung der Flags

Bei der Abarbeitung einiger Applikationsanweisungen werden automatisch von der SPS verschiedene Flags (Sondermerker) gesetzt bzw. zurückgesetzt. Ein gesetztes Flag zeigt einen bestimmten Programmzustand an (z. B. zulässiger numerischer Datenbereich bei der Ausführung einer Anweisung überschritten).

Diese Flags werden jedesmal gesetzt bzw. zurückgesetzt, wenn die entsprechende Anweisung im Programm aktiviert wird. Das Setzen oder Zurücksetzen eines Flags hat jedoch nicht zur Folge, daß im nächsten Programmzyklus, wenn diese Anweisung nicht ausgeführt wird, das Flag seinen Status ändert.

Eine Übersicht über alle Flags und ihre Bedeutung finden Sie in Abs. 10.1.3.

6.1.8 Programmablauffehler bei der Ausführung von Applikationsanweisungen

Bei fehlerhaft programmierten Applikationsanweisungen oder Operandenadressen wird eine Fehlermeldung in einem Fehlerdatenregister abgespeichert.

Das Kapitel enthält eine detaillierte Übersicht sämtlicher Fehlerdatenregister und Fehlermeldungen.

6.1.9 32-Bit-Anweisungen

Wenn eine Anweisung als 32-Bit-Anweisung ausgeführt werden soll, wird der Anweisung der Parameter „D“ vorangestellt. Ob eine Anweisung als 32-Bit-Anweisung ausführbar ist, kann anhand der Übersichtstabelle zu jeder Applikationsanweisung ersehen werden.

Bei der Verarbeitung von Applikationsanweisungen mit Wortoperanden müssen Sie darauf achten, daß zwei Wortoperanden zu einem 32-Bit-Wort zusammengefaßt werden, aber immer das niedrigstwertige Byte adressiert wird (untere 16 Bit).

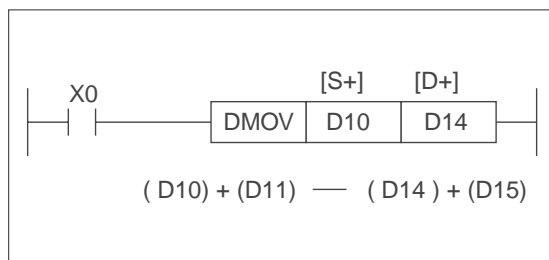


Abb. 6-6:

Programmierbeispiel zum Einsatz der 32-Bit-Anweisung DMOV

C000209C

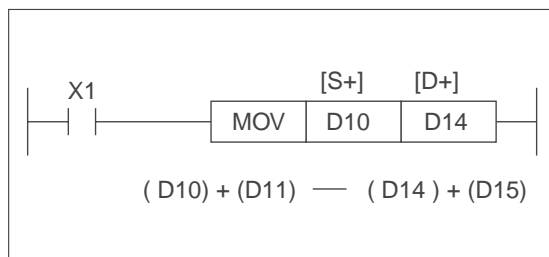


Abb. 6-7:

Programmierbeispiel zum Einsatz der 16-Bit-Anweisung MOV

C000210C

HINWEIS

Beim Einsatz der Indexregister in Zusammenhang mit einer 32-Bit-Anweisung dürfen Sie nur das Indexregister Z adressieren.

6.1.10 Übersicht der Applikationsanweisungen

Einteilung	Anweisung	FNC	Bedeutung	Referenz	Steuerung		
					FX1S	FX1N	FX2N
Programmablaufanweisungen	CJ	00	Sprung innerhalb eines Programms	6.2.1	●	●	●
	CALL	01	Aufruf eines Unterprogramms	6.2.2	●	●	●
	SRET	02	Ende eines Unterprogramms	6.2.3	●	●	●
	IRET	03	Interrupt-Programm abschließen	6.2.4	●	●	●
	EI	04	Interrupt-Programm aktivieren	6.2.4	●	●	●
	DI	05	Interrupt-Programm deaktivieren	6.2.4	●	●	●
	FEND	06	Ende eines Programmbereichs	6.2.5	●	●	●
	WDT	07	Watch-Dog-Timer auffrischen	6.2.6	●	●	●
	FOR	08	Anfang einer Programmwiederholung	6.2.7	●	●	●
	NEXT	09	Ende einer Programmwiederholung	6.2.7	●	●	●
Vergleichs- und Transferanweisungen	CMP	10	Numerische Daten vergleichen	6.3.1	●	●	●
	ZCP	11	Numerische Datenbereiche vergleichen	6.3.2	●	●	●
	MOV	12	Datentransfer	6.3.3	●	●	●
	SMOV	13	Shift-Transfer	6.3.4			●
	CML	14	Kopieren und invertieren	6.3.5			●
	BMOV	15	Block-Transfer	6.3.6	●	●	●
	FMOV	16	Transfer von gleichen Daten	6.3.7			●
	XCH	17	Austausch von Daten	6.3.8			●
	BCD	18	BCD-Konvertierung	6.3.9	●	●	●
	BIN	19	Binär-Konvertierung	6.3.10	●	●	●
Arithmetische Anweisungen	ADD	20	Addition numerischer Daten	6.4.1	●	●	●
	SUB	21	Subtraktion numerischer Daten	6.4.2	●	●	●
	MUL	22	Multiplikation numerischer Daten	6.4.3	●	●	●
	DIV	23	Division numerischer Daten	6.4.4	●	●	●
	INC	24	Inkrementieren	6.4.5	●	●	●
	DEC	25	Dekrementieren	6.4.6	●	●	●
	AND	26	Logische UND-Verknüpfung	6.4.7	●	●	●
	OR	27	Logische ODER-Verknüpfung	6.4.8	●	●	●
	XOR	28	Logische Exklusiv-ODER-Verknüpfung	6.4.9	●	●	●
	NEG	29	Negation von Daten	6.4.10			●

Tab. 6-7: Übersicht der Applikationsanweisungen (1)

Einteilung	Anweisung	FNC	Bedeutung	Referenz	Steuerung		
					FX1S	FX1N	FX2N
Verschiebe- anweisungen	ROR	30	Rotation nach rechts	6.5.1			●
	ROL	31	Rotation nach links	6.5.2			●
	RCR	32	Rotieren von Bits nach rechts	6.5.3			●
	RCL	33	Rotieren von Bits nach links	6.5.4			●
	SFTR	34	Binäre Daten bitweise verschieben, rechts	6.5.5	●	●	●
	SFTL	35	Binäre Daten bitweise verschieben, links	6.5.5	●	●	●
	WSFR	36	Daten wortweise nach rechts verschieben	6.5.6			●
	WSFL	37	Daten wortweise nach links verschieben	6.5.7			●
	SFWR	38	Schreiben in einen FIFO-Speicher	6.5.8	●	●	●
	SFRD	39	Lesen aus einem FIFO-Speicher	6.5.9	●	●	●
Datenoperationen	ZRST	40	Operandenbereiche zurücksetzen	6.6.1	●	●	●
	DECO	41	Daten decodieren	6.6.2	●	●	●
	ENCO	42	Daten codieren	6.6.3	●	●	●
	SUM	43	Ermittlung gesetzter Bits	6.6.4			●
	BON	44	Überprüfen eines Bits	6.6.5			●
	MEAN	45	Ermittlung von Durchschnittswerten	6.6.6			●
	ANS	46	Starten eines Zeitintervalls	6.6.7			●
	ANR	47	Rücksetzen von Anzeige-Bits	6.6.8			●
	SQR	48	Ermittlung der Quadratwurzel	6.6.9			●
	FLT	49	Umwandlung des Zahlenformats	6.6.10			●
High-Speed- Anweisungen	REF	50	Ein- und Ausgänge auffrischen	6.7.1	●	●	●
	REFF	51	Einstellen der Eingangsfiler	6.7.2			●
	MTR	52	Einlesen einer Matrix (MTR)	6.7.3			●
	DHSCS	53	Setzen durch High-Speed-Counter	6.7.4	●	●	●
	DHSCR	54	Rücksetzen durch High-Speed-Counter	6.7.4	●	●	●
	DHSZ	55	Bereichsvergleich	6.7.5			●
	SPD	56	Geschwindigkeitserkennung	6.7.6	●	●	●
	PLSY	57	Impulsausg. einer def. Anzahl von Impulsen	6.7.7	●	●	●
	PWM	58	Impulsausg. mit Impulsweitenmodulation	6.7.8	●	●	●
	PLSR	59	Ausgabe einer bestimmten Anzahl von Impulsen	6.7.9	●	●	●
Anwendungs- bezogene Anweisungen	IST	60	Schrittstatus initialisieren	6.8.1	●	●	●
	SER	61	Suchanweisung	6.8.2			●
	ABSD	62	Absoluter Counter-Vergleich	6.8.3	●	●	●
	INCD	63	Inkrementaler Counter-Vergleich	6.8.4	●	●	●
	TTMR	64	Teaching-Timer	6.8.5			●
	STMR	65	Sonder-Timer	6.8.6			●
	ALT	66	Flip-Flop-Funktion	6.8.7	●	●	●
	RAMP	67	Rampenfunktion	6.8.8	●	●	●
	ROTC	68	Rundtisch-Positionierung	6.8.9			●
	SORT	69	Sortieranweisung	6.8.10			●

Tab. 6-7: Übersicht der Applikationsanweisungen (2)

HINWEIS

Die Applikationsanweisungen FNC 70 – 98 werden im Kapitel 7 beschrieben.

6.2 Programmablaufanweisungen

Übersicht der Anweisungen FNC 00 bis 09

Symbol	FNC	Bedeutung	Abschnitt
CJ	00	Sprung innerhalb eines Programms	6.2.1
CALL	01	Aufruf eines Unterprogramms	6.2.2
SRET	02	Ende eines Unterprogramms	6.2.3
IRET	03	Interrupt-Programm abschließen	6.2.4
EI	04	Interrupt-Programm aktivieren	6.2.4
DI	05	Interrupt-Programm deaktivieren	6.2.4
FEND	06	Ende eines Programmbereichs	6.2.5
WDT	07	Watch-Dog-Timer auffrischen	6.2.6
FOR	08	Anfang einer Programmwiederholung	6.2.7
NEXT	09	Ende einer Programmwiederholung	6.2.7

Tab. 6-8: Übersicht der Anweisungen FNC 00 bis 09

6.2.1 Sprung innerhalb eines Programms (CJ)

	CJ		FNC 00		
	Sprung innerhalb eines Programms				
	CPU	FX1S	FX1N	FX2N	
		●	●	●	
Operanden	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
Pointer P0 bis P63 (Indizierung ist zulässig) P63 bedeutet Sprung zur END-Anweisung	●	16 Bit	32 Bit	CJ / CJP	3
		●		Pointer P**	1

Funktionsweise

Mit der CJ-Anweisung können Teile eines Programms übersprungen werden. Die Programmzykluszeit kann durch Einsatz der CJ-Anweisung reduziert werden.

Beschreibung

- Das Sprungziel wird durch Festlegen einer Markierung (Pointer-Markierung) im Programm definiert.
- Die Angabe der Sprungzieladresse (Pointer-Adresse) legt fest, zu welcher Pointer-Markierung der Sprung erfolgen soll.
- Wird innerhalb der Sprungroutine eine Rücksetzanweisung für remanente Counter programmiert, ist der Rücksetzvorgang (Löschen von Istwerten) auch dann noch wirksam, wenn der Strompfad der Counter-Spule übersprungen wird.
- Es besteht die Möglichkeit, Ausgänge doppelt zu belegen.

HINWEIS

Achten Sie bei einer Doppelbelegung von Ausgängen darauf, daß nie beide Ausgänge zur gleichen Zeit aktiv sein können. Dies würde zu Programmablaufstörungen führen.

Festlegen der Pointer-Markierung im Programm

- Die Pointer-Markierung wird bei der Programmierung in Anweisungsliste direkt vor einem Strompfad (vor einer LD- oder LDI-Anweisung) festgelegt.
- Bei der Programmierung in Kontaktplan wird die Pointer-Markierung links vor dem Strompfad festgelegt.

Beispiel ▾

Einsatz der CJ-Anweisung

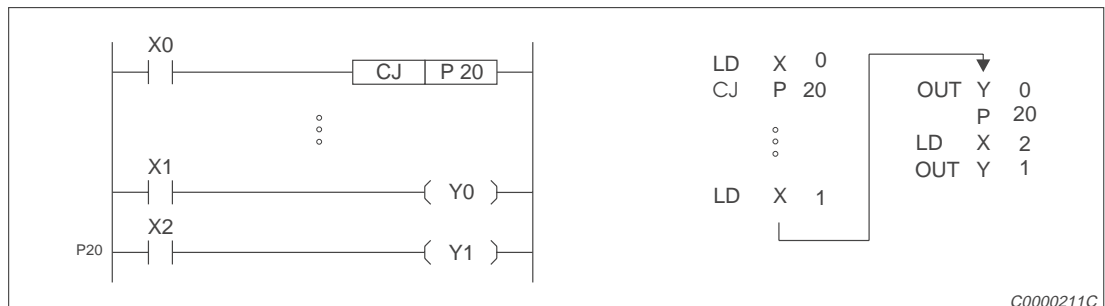


Abb. 6-8: Programmierbeispiel zur CJ-Anweisung

Ist X0 eingeschaltet, findet ein Sprung zur Pointer-Markierung P20 statt.



Zweimaliger Einsatz der Sprungzieladresse (Pointer-Adresse) in einem Programm

Beispiel ▾

Zweimaliger Einsatz der Pointer-Adresse P9 in einem Programm.

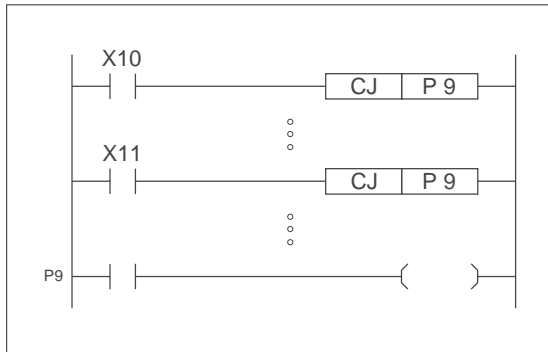


Abb. 6-9:
 Programmierbeispiel für den zweimaligen Einsatz der Pointer-Adresse P9 in einem Programm

C000212C

Ist X10 eingeschaltet, findet ein Sprung zur Pointermarkierung P9 statt. Ist X10 ausgeschaltet und X11 eingeschaltet, wird ebenso ein Sprung nach P9 ausgeführt. △

HINWEIS

Die gleiche Pointer-Markierung darf jedoch nicht mehrfach in einem Programm benutzt werden. Es tritt sonst ein Programmablauffehler auf.

Festlegen der Pointer-Markierung vor der CJ-Sprunganweisung

Ein Rücksprung kann auch innerhalb des Programms ausgeführt werden.

HINWEIS

Wenn das Eingangssignal für die CJ-Anweisung länger als 200 ms ansteht, tritt ein Watch-Dog-Timer-Fehler auf.

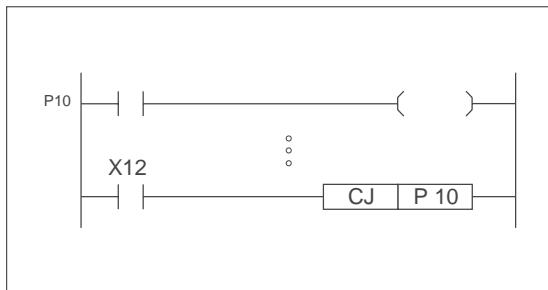


Abb. 6-10:
 Programmierbeispiel zum Festlegen einer Pointer-Markierung vor der CJ-Sprunganweisung

C000213C

Sprünge im Master-Control-Bereich

Den Programmablauf beim Einsatz der CJ-Anweisungen in Zusammenhang mit den MC- und MCR-Anweisungen zeigt das folgende Beispiel:

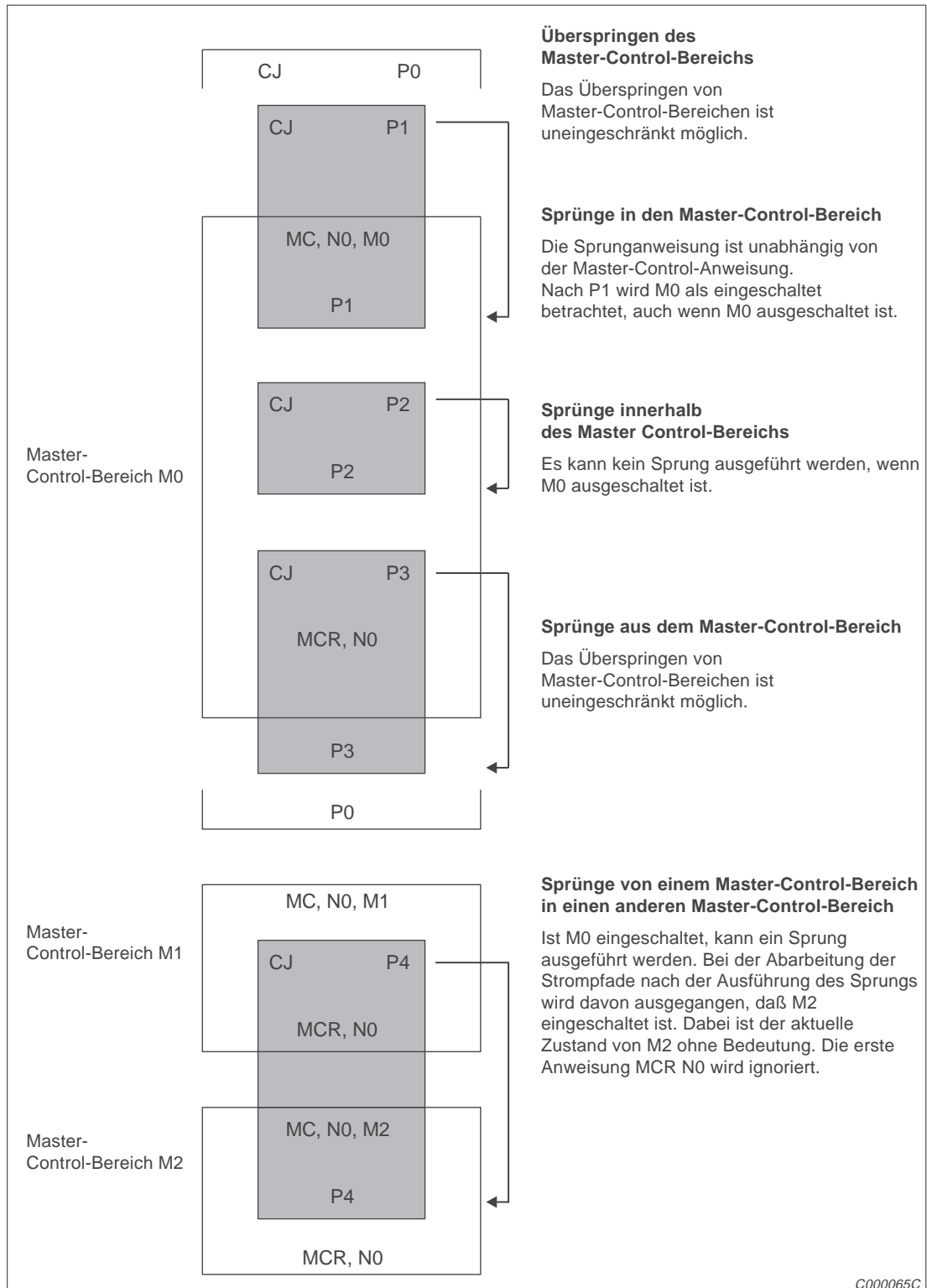


Abb. 6-11: Beispiel zu Sprüngen im Master-Control-Bereich

Verhalten der Kontakte und Spulen innerhalb eines übersprungenen Programmteils

Operanden	Status des Kontaktes und der Spule vor dem Sprung	Status des Kontaktes und der Spule nach dem Sprung	Bemerkung
Ausgänge Y	EIN	EIN	—
Merker M	EIN	EIN	—
Schrittstatus S	EIN	EIN	—
Timer T	EIN	EIN	Der Zeitablauf wird gestoppt. Der aktuelle Zeitwert wird gespeichert. Ist die Sprungbedingung nicht mehr erfüllt, wird der Zeitablauf fortgesetzt.
Counter C	EIN	EIN	Die Zählung wird gestoppt. Der aktuelle Zählerwert wird gespeichert. Ist die Sprungbedingung nicht mehr erfüllt, wird die Zählung fortgesetzt.
Applikationsanweisungen	—	—	Die Ausführung wird gestoppt. Die Anweisungen, wie z. B. RAMP, INC oder DEC, behalten jedoch ihre aktuellen Datenwerte.

Tab. 6-9: Verhalten der Kontakte und Spulen bei der Abarbeitung einer Sprunganweisung

6.2.2 Aufruf eines Unterprogramms (CALL)

		CALL		FNC 01	
		Aufruf eines Unterprogramms			
		CPU	FX1S	FX1N	FX2N
			●	●	●
Operanden	D	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	Pointer P0 bis P62 (Indizierung ist erlaubt)	●	16 Bit	32 Bit	CALL/CALLP 3 Pointer P** 1

Funktionsweise

Mit der CALL-Anweisung werden Unterprogramme aufgerufen.

Beschreibung

- Ein Unterprogramm wird mit einem Pointer (P0 bis P62) markiert und über die CALL-Anweisung aufgerufen.
- Am Ende eines Unterprogramms muß eine SRET-Anweisung stehen.
- Unterprogramme werden hinter der FEND-Anweisung und vor der END-Anweisung programmiert.
- Wenn eine CALL-Anweisung aktiviert wird, findet ein Sprung zu der angegebenen Pointer-Markierung statt. Nach der Bearbeitung der SRET-Anweisung findet ein Rücksprung zu der auf die CALL-Anweisung folgenden Anweisung statt.
- In einem Unterprogramm aktivierte Operanden bleiben nach der Abarbeitung des Unterprogramms bis zur erneuten Bearbeitung des Unterprogramms aktiviert.
- In einem Unterprogramm sollten die Timer T192 bis T199 und T246 bis T249 genutzt werden.
- Derselbe Pointer kann in beliebig vielen CALL-Anweisungen genutzt werden. Er darf aber nur einmal als Pointer-Markierung programmiert werden.

HINWEIS

Innerhalb eines Unterprogramms können weitere Unterprogramme aufgerufen werden. Es sind maximal 4 Verzweigungsebenen möglich.

6.2.3 Ende eines Unterprogramms (SRET)

		SRET		FNC 02	
		Aufruf eines Unterprogramms			
Operanden		CPU	FX1S	FX1N	FX2N
			●	●	●
Operanden	D	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	—		●	16 Bit	
			●		SRET
					1

Funktionsweise

Mit der SRET-Anweisung wird das Ende eines Unterprogramms gekennzeichnet.

Beschreibung

- Ein Unterprogramm wird mit einem Pointer (P0 bis P62) markiert und über die CALL-Anweisung aufgerufen.
- Am Ende eines Unterprogramms muß eine SRET-Anweisung stehen.
- Unterprogramme werden hinter der FEND-Anweisung und vor der END-Anweisung programmiert.
- Nach der Bearbeitung der SRET-Anweisung findet ein Rücksprung zu der auf die CALL-Anweisung folgenden Anweisung statt.

HINWEIS Eine SRET-Anweisung kann nur im Zusammenhang mit der CALL-Anweisung programmiert werden.

Beispiel ▾

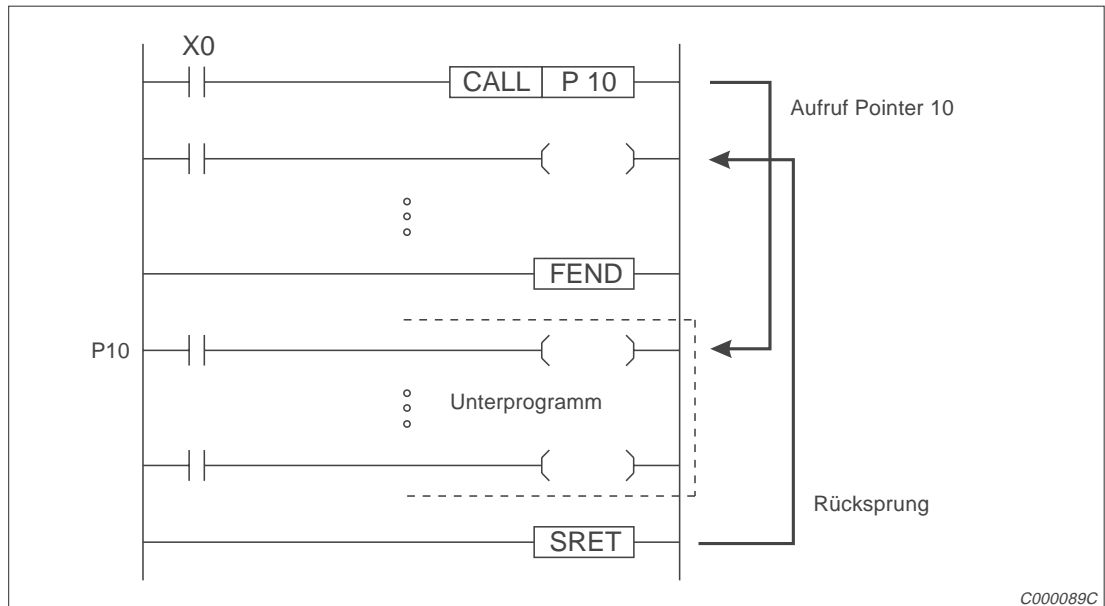


Abb. 6-12: Programmierbeispiel zum Einsatz der CALL- und SRET-Anweisungen



6.2.4 Einsatz eines Interrupt-Programms (IRET, EI, DI)

		IRET		FNC 03			
		Interrupt-Programm abschließen					
Operanden		D	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				CPU	FX1S	FX1N	FX2N
		—		●	●	●	
				16 Bit	32 Bit	IRET	1
						Pointer I***	1

		EI		FNC 04			
		Pulse-Catch-/Interrupt-Programm aktivieren					
Operanden		D	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				CPU	FX1S	FX1N	FX2N
		—		●	●	●	
				16 Bit	32 Bit	EI	1
						Pointer I***	1

		DI		FNC 05			
		Interrupt-Programm deaktivieren					
Operanden		D	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				CPU	FX1S	FX1N	FX2N
		—		●	●	●	
				16 Bit	32 Bit	DI	1
						Pointer I***	1

Funktionsweise

Aufruf, Abschließen, Aktivieren und Deaktivieren eines Interrupt-Programms

Interrupt-Programm aufrufen

- Beim Aufruf eines Interrupt-Programms wird das SPS-Programm verlassen, und es erfolgt ein Sprung zum Interrupt-Programm. Nachdem das Interrupt-Programm beendet ist, erfolgt wieder ein Sprung zum SPS-Programm.
- Der Beginn eines Interrupt-Programms wird durch Festlegen einer Markierung (Interrupt-Pointer) definiert.
- Das Ende eines Interrupt-Programms wird mit der IRET-Anweisung festgelegt.
- Die Eingänge X0 bis X5 bilden die Interrupt-Eingänge.
- Die Interrupt-Signale müssen über eine Pulsweite von mindestens 200 µs verfügen.
- Ein Interrupt-Programm muß am Ende eines SPS-Programms hinter der letzten FEND-Anweisung und vor der END-Anweisung programmiert werden.

FX2N

- Timer-Interrupts sind möglich.
- Counter-Interrupts sind möglich.

HINWEIS

Die Eingänge X0 bis X5 können nicht gleichzeitig zur Verarbeitung von Interrupt-Signalen und zur Verarbeitung von High-Speed-Zählersignalen eingesetzt werden.

Interrupt-Pointer adressieren

- Die Adressierung eines Interrupt-Pointers müssen Sie wie folgt vornehmen:

Interrupt-Pointer: I ◆ 0 ◇

◆: Adresse 0 bis 5; entspricht Eingang X0 bis X5

◇: 0:= Interrupt bei abfallender Eingangssignalfanke
1:= Interrupt bei ansteigender Eingangssignalfanke

FX2N**Timer-Interrupt (nur FX2N)**

- Das Interrupt-Programm wird jeweils nach Ablauf der vorgegebenen Zeit bearbeitet.

Interrupt-Pointer: I ◆◇◇

◇◇: Intervallzeit 10 bis 99 ms

◆: Adresse 6 bis 8 (T6 bis T8)

FX2N**Counter-Interrupt (nur FX2N)**

- Das Interrupt-Programm wird nach Erreichen eines vorgegebenen Zählwertes bearbeitet.

Interrupt-Pointer: I 0 ◇ 0

◇: Interrupt-Nummer 1 bis 6

HINWEIS

| Jede Adresse darf nur einmal verwendet werden.

Einsatz der EI- und DI-Anweisung

- Mit der **EI-Anweisung** können Sie ein **Interrupt-Programm aktivieren**. Dies bedeutet, daß nach der Abarbeitung der EI-Anweisung, Signalwechsel, die an einem der Eingänge X0 bis X5 auftreten, als Interrupt-Signale im Programm verarbeitet werden.
- Mit der **DI-Anweisung** können Sie ein **Interrupt-Programm deaktivieren**. Dies bedeutet, daß nach Abarbeitung der DI-Anweisung, Signalwechsel, die an einem der Eingänge X0 bis X5 auftreten, nicht mehr als Interrupt-Signale im Programm verarbeitet werden.

HINWEIS

| Wenn keine der beiden Anweisungen EI oder DI programmiert wurde, ist der Interrupt-Modus nicht aktiviert, d. h., es können dann keine Interrupt-Signale verarbeitet werden.

Abarbeitung von Interrupt-Programmen

- Während der Ausführung eines Interrupt-Programms können keine anderen Interrupt-Programme aufgerufen werden. Sie können jedoch zwei Verzweigungsebenen programmieren. Die Anweisungen EI und DI müssen Sie dann innerhalb des Interrupt-Programms einsetzen.
- Mehrere aufeinanderfolgende Interrupt-Programme werden in der Reihenfolge ihres Aufrufs abgearbeitet.
- Werden mehrere Interrupt-Programme gleichzeitig aufgerufen, wird das Interrupt-Programm mit der niedrigsten Pointer-Adresse zuerst abgearbeitet.
- Ein Interrupt-Programm, das in einem Bereich zwischen einer DI-Anweisung und einer EI-Anweisung aufgerufen wurde, wird erst nach der Ausführung der EI-Anweisung abgearbeitet.

Ausschalten beliebiger Interrupts

- Sie können beliebige Interrupts durch Einschalten der zugehörigen Sondermerker zeitweilig oder permanent Ausschalten. Die entsprechenden Sondermerker werden in Kapitel 6 angegeben. Für alle Steuerungen ist der erste Sondermerker M8050, der den Interrupt I0^{①②} ausschaltet.

HINWEISE

Setzen Sie nie einen Sondermerker, ohne sich seiner Funktion sicher zu sein. Nicht alle Steuerungen arbeiten immer mit den gleichen Sondermerkern.

High-Speed-Counter-Interrupts können immer nur als einzelne Gruppe mit dem Sondermerker M8059 ausgeschaltet werden.

Es können maximal zwei Verzweigungsebenen programmiert werden.

Ein Interrupt-Programm wird nicht ausgeführt, wenn der zugehörige Sondermerker aktiviert ist. So kommt das Interrupt-Programm I□ ** nicht zur Ausführung, wenn der Sondermerker M805□ (□: 1, 2, 3, 4, 5) aktiviert ist.

Speichern der Signalwechsel der Interrupt-Eingänge

Diese Funktion (Pulse-Catch-Funktion) ermöglicht das Speichern der Signalwechsel der Interrupt-Eingänge X0 bis X5 in den Sondermerkern M8170 – M8175. Diese Speicherfunktion kann nur einmal für einen Eingang gleichzeitig ausgeführt werden. Die Pulse-Catch-Funktion wird mit der EI-Anweisung aktiviert.

Beispiel ▾

Interrupt-Pointer adressieren

Pointer: I001

Erklärung: Interrupt-Eingang X0, Interrupt bei ansteigender Eingangssignalfanke (Signalwechsel von „0“ auf „1“) △

Beispiel ▾ Einsatz der Anweisungen EI, DI und IRET

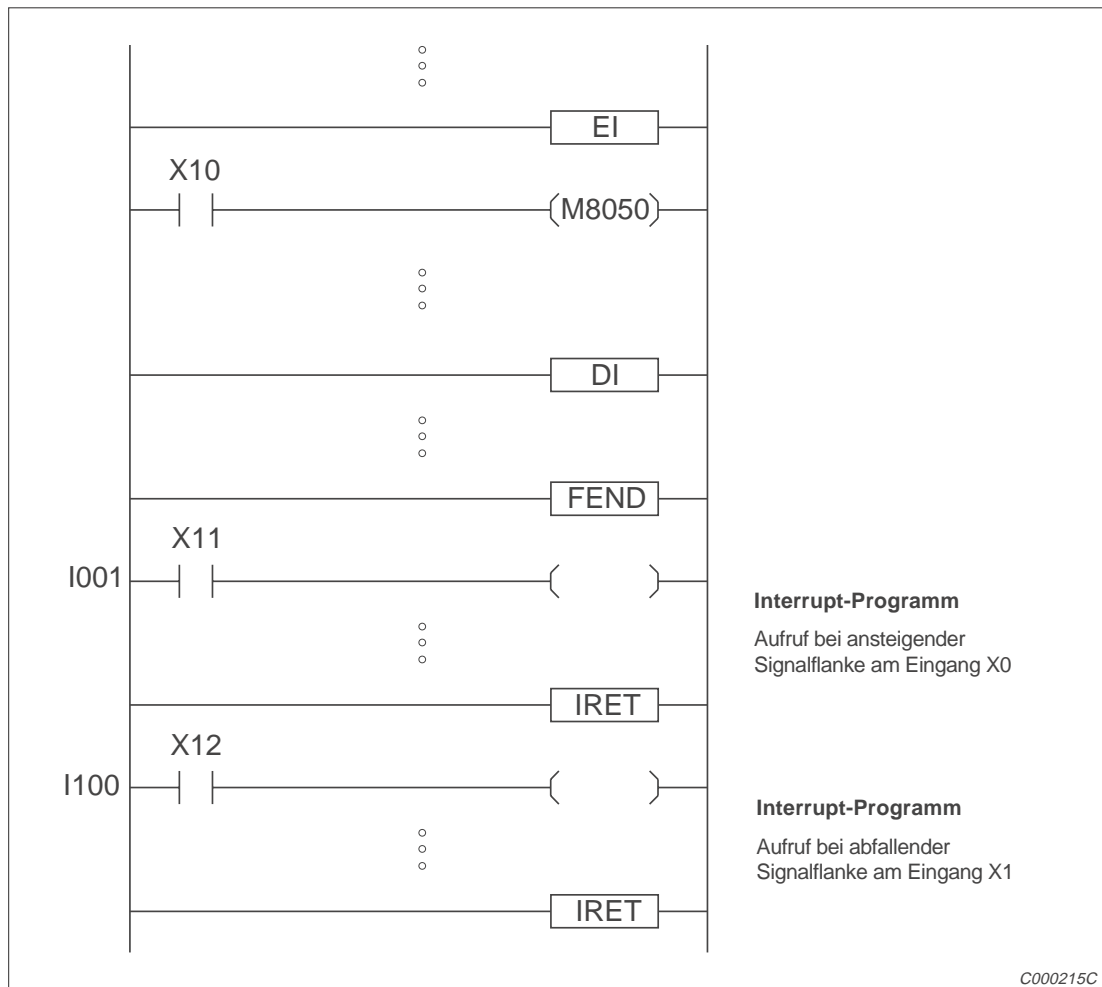


Abb. 6-13: Programmierbeispiel zum Einsatz der Anweisungen EI, DI und IRET

- Wenn am Eingang X0 ein Interrupt-Signal ansteht, während ein Programmschritt innerhalb des Bereichs von der EI-Anweisung bis zur DI-Anweisung ausgeführt wird, findet ein Sprung zum Interrupt-Programm I001 statt. Das Interrupt-Programm wird ausgeführt, und es erfolgt ein Rücksprung zum SPS-Programm.
- Das Interrupt-Programm I001 wird nicht ausgeführt, wenn der Sondermerker M8050 aktiviert ist (Eingang X10 eingeschaltet).
- Wenn am Eingang X1 ein Interrupt-Signal ansteht, während ein Programmschritt innerhalb des Bereichs von der EI-Anweisung bis zur DI-Anweisung ausgeführt wird, findet ein Sprung zum Interrupt-Programm I100 statt. Das Interrupt-Programm wird ausgeführt, und es erfolgt ein Rücksprung zum SPS-Programm.
- Wenn beide Signale X0 und X1 gleichzeitig auftreten, wird erst das Interrupt-Programm I001 und danach das Interrupt-Programm I100 bearbeitet.

△

6.2.5 Ende eines Programmbereichs (FEND)

		FEND		FNC 06	
		Ende eines Programmbereichs			
Operanden	D	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	—		16 Bit	32 Bit	
			FEND	1	

Funktionsweise

Beenden von einzelnen Programmbereichen innerhalb eines SPS-Programms

Beschreibung

- Mit der FEND-Anweisung wird das Ende eines Programmbereichs festgelegt. Sie können mehrere FEND-Anweisungen innerhalb eines SPS-Programms einsetzen.
- Nach Abarbeitung der FEND-Anweisung erfolgt die Ausgangsbearbeitung. Danach findet ein Rücksprung zum Programmschritt 0 statt. Die Eingangsbearbeitung und der Watch-Dog-Timer werden aufgefrischt.

HINWEISE

Programmieren Sie die Interrupt-Programme zwischen der letzten FEND-Anweisung und der END-Anweisung.

Verwechseln Sie nicht die FEND-Anweisung mit der END-Anweisung. Mit der END-Anweisung wird das gesamte SPS-Programm abgeschlossen (siehe Abs. 4.13).

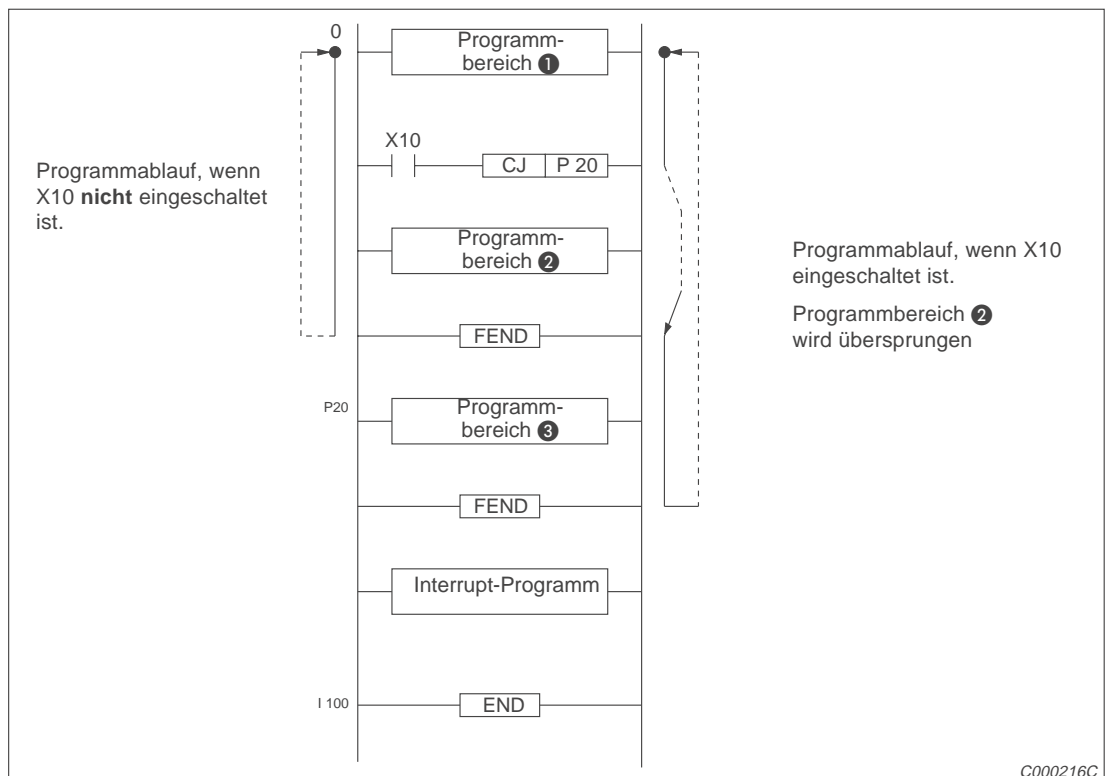


Abb. 6-14: Programmierbeispiel zum Einsatz der FEND-Anweisung

6.2.6 Watch-Dog-Timer auffrischen (WDT)

		WDT		FNC 07		
		Watch-Dog-Timer auffrischen				
Operanden		CPU	FX1S	FX1N	FX2N	
			●	●	●	
	D	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	—	●	16 Bit	32 Bit	WDT	1
			●		WDTP	1

Funktionsweise

Mit Hilfe der WDT-Anweisung können Sie längere Programme in einzelne Programmabschnitte unterteilen. Die Programmzykluszeit wird für jeden einzelnen Programmabschnitt von der Steuerung ermittelt (Watch-Dog-Timer wird nach jedem Programmabschnitt aufgefrischt). Mit der WDT-Anweisung ist es möglich, Programme zu verarbeiten, deren Programmzykluszeit 200 ms überschreitet.

Beschreibung

- Die WDT-Anweisung müssen Sie einsetzen, wenn die Programmzykluszeit vom Programmschritt 0 bis zur END- oder FEND-Anweisung den Wert von **200 ms** überschreitet.
- Des weiteren kann die WDT-Anweisung nach einer Pointer-Markierung programmiert werden, wenn sich diese im Programm vor der zugehörigen Sprunganweisung (CJ-Anweisung) befindet (siehe Abs. 6.2.1).
- Die WDT-Anweisung kann auch innerhalb einer FOR-NEXT-Schleife eingesetzt werden (siehe Abs. 6.2.7).

HINWEIS

Der Watch-Dog-Timer wird bei jeder Ausführung der Anweisung END, FEND oder WDT aufgefrischt.

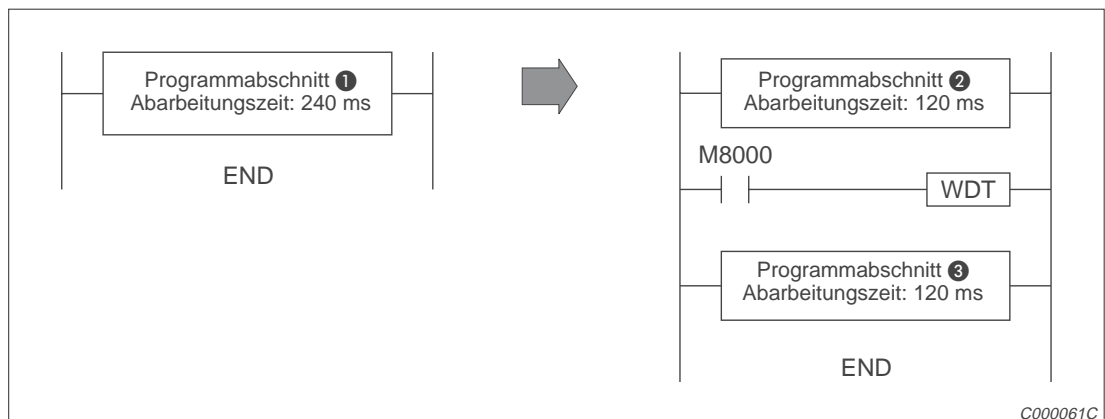


Abb. 6-15: Programmierbeispiel zum Einsatz der WDT-Anweisung

Die Abarbeitungszeit für den Programmabschnitt 1 überschreitet den Wert von 200 ms. Der Programmabschnitt 1 wurde deshalb durch Einsatz der WDT-Anweisung in zwei Programmabschnitte (2,3) aufgeteilt. Die Programmabschnitte 2 und 3 benötigen jeweils nur 120 ms Abarbeitungszeit. △

Programmzykluszeitwert im Sonderregister D8000 verändern

Überschreitet die Programmzykluszeit wiederholt den Wert von 200 ms, können Sie den Wert der maximal zulässigen Programmzykluszeit im Sonderregister D8000 verändern.

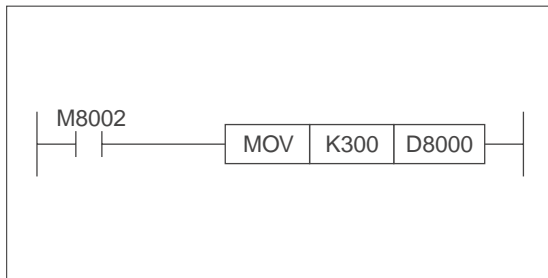


Abb. 6-16:

Programmierbeispiel zur Einstellung einer maximal zulässige Programmzykluszeit im Datenregister D8000 auf den Wert 300 ms.

C000070C

6.2.7 Programmteile wiederholen (FOR, NEXT)

		FOR		FNC 08		
		Anfang einer Programmwiederholung				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	S	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		16 Bit	32 Bit	FOR	3
			●			

		NEXT		FNC 09		
		Ende einer Programmwiederholung				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	S	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	—		16 Bit	32 Bit	NEXT	1
			●			

Funktionsweise

Programmieren von Programmwiederholungen (Programmschleifen)

Beschreibung

- Der Programmteil zwischen der FOR- und der NEXT-Anweisung wird n Mal wiederholt. Anschließend werden die Programmschritte nach der NEXT-Anweisung ausgeführt.
- Der Wert n muß innerhalb des folgenden Bereichs liegen: n: +1 bis +32 767. Wird für n ein Wert zwischen 0 und -32 767 angegeben, wird die FOR-NEXT-Schleife nur einmal abgearbeitet.
- Sie können bis zu fünf FOR-NEXT-Verzweigungsebenen programmieren.

HINWEIS

FOR- und NEXT-Anweisungen dürfen nur paarweise programmiert werden. Zu jeder FOR-Anweisung muß eine passende NEXT-Anweisung programmiert werden.

Fehlerquellen

In den folgenden Fällen tritt ein Programmablauffehler auf:

- Eine NEXT-Anweisung wurde vor einer FOR-Anweisung programmiert.
- Eine NEXT-Anweisung wurde nach der FEND- oder END-Anweisung programmiert.
- Die Anzahl der NEXT-Anweisungen stimmt nicht mit der Anzahl der FOR-Anweisungen überein.

Beispiel ▾ | Einsatz der FOR-, NEXT-Anweisungen

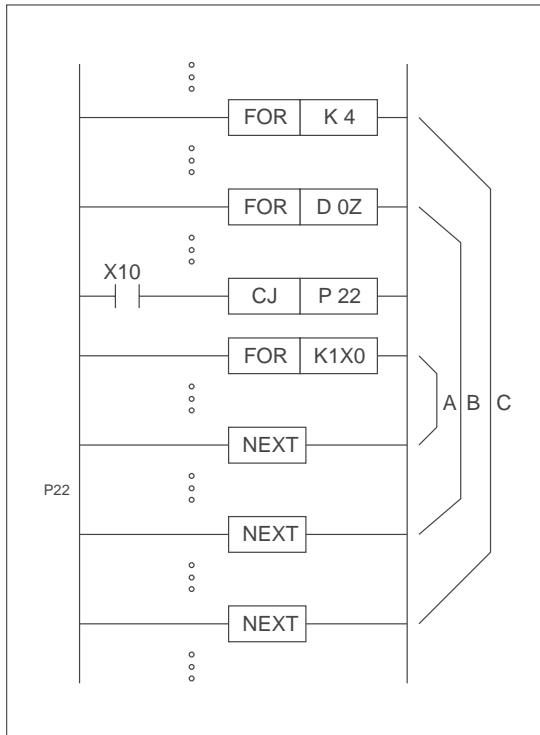


Abb. 6-17:
 Programmierbeispiel zum Einsatz der FOR-,
 NEXT-Anweisungen

C000018C

In dem Beispiel wurden drei ineinanderverschachtelte FOR-NEXT-Verzweigungsebenen programmiert.

- Der Programmabschnitt C wird viermal abgearbeitet. Anschließend werden die Programmschritte nach der dritten NEXT-Anweisung ausgeführt.
- Bei jeder Ausführung des Abschnitts C wird der Programmabschnitt B sechsmal abgearbeitet, wenn im Datenregister D0Z der Wert 6 steht.
- Der Abschnitt B wird demnach 24 Mal abgearbeitet.
- Wenn der Eingang X10 eingeschaltet ist, wird die FOR-NEXT-Schleife A mit Hilfe der CJ-Anweisung übersprungen.
- Wenn X10 ausgeschaltet ist und der Inhalt von K1X0 gleich 7 ist, wird bei jeder Ausführung des Abschnitts B der Programmabschnitt A siebenmal abgearbeitet.
- Der Abschnitt A wird demnach insgesamt 168 (4 x 6 x 7) Mal abgearbeitet.

△

6.3 Vergleichs- und Transferanweisungen

Übersicht der Anweisungen FNC 10 bis 19

Symbol	FNC	Bedeutung	Abschnitt
CMP	10	numerische Daten vergleichen	6.3.1
ZCP	11	numerische Datenbereiche vergleichen	6.3.2
MOV	12	Datentransfer	6.3.3
SMOV	13	Shift-Transfer	6.3.4
CML	14	Kopieren und invertieren	6.3.5
BMOV	15	Block-Transfer	6.3.6
FMOV	16	Transfer von gleichen Daten	6.3.7
XCH	17	Austausch von Daten	6.3.8
BCD	18	BCD-Konvertierung	6.3.9
BIN	19	Binär-Konvertierung	6.3.10

Tab. 6-10: Übersicht der Anweisungen FNC 10 bis 19

6.3.1 Numerische Daten vergleichen (CMP, DCMP)

		CMP		FNC 10										
		Numerische Daten vergleichen												
CPU		FX1S		FX1N		FX2N								
		●		●		●								
Operanden	S+, S2+		D+		Puls-Anweisung (P)		Verarbeitung		Programmschritte					
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		Y, M, S		●		16 Bit		32 Bit		CMP/CMPP		7	
							●		●		DCMP/DCMPP		13	

Funktionsweise

Vergleich zwischen zwei numerischen Datenwerten (größer, kleiner, gleich)

Beschreibung

- Die Daten in den beiden Quellen (S1+) und (S2+) werden miteinander verglichen.
- Das Ergebnis des Vergleichs (größer, gleich, kleiner) wird durch Setzen eines Merkers M, Schrittstatusoperanden S oder Ausgangs Y angezeigt. Die Festlegung, welcher Operand gesetzt werden soll, erfolgt an der Zieladresse (D+).

$(S1+) > (S2+) \rightarrow (D+)$

$(S1+) = (S2+) \rightarrow ((D+)+1)$

$(S1+) < (S2+) \rightarrow ((D+)+2)$

- Die Daten in S1+ und S2+ werden als binäre Daten behandelt.

Fehlerquellen

- Die CMP-Anweisung erfordert die Angabe von drei Operanden. Werden weniger Operanden angegeben, tritt der Fehler mit dem Fehlercode 6503 auf. Die Ausführung der Anweisung wird unterbrochen.
- Wird ein nicht zulässiger Operand programmiert, wird ein Fehler mit dem Fehlercode 6705 ausgegeben. Als Zieladresse dürfen z.B. keine Eingänge X, Datenregister D, Timer T oder Counter C angegeben werden.
- Wenn der angegebene Operand den für ihn zulässigen Bereich überschreitet, wird ein Fehler mit dem Fehlercode 6706 ausgegeben. Fehler dieser Art treten z.B. bei zu großen Zahlenwerten in Verbindung mit der Index-Adressierung auf.

Eine detaillierte Beschreibung der Fehlercodes enthält Kapitel 11.

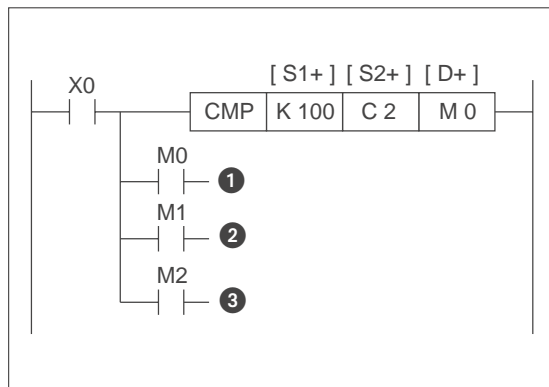
Beispiel ▾ Einsatz der CMP-Anweisung

Abb. 6-18:
 Programmierbeispiel zum Einsatz der CMP-
 Anweisung

C000071C

In der Zieladresse (D+) ist im Beispiel der Merker M0 angegeben. Entsprechend dem Ergebnis des Vergleichs werden demnach die Merker M0, M1 und M2 wie folgt geschaltet:

- ① M0: EIN, wenn $K100 > \text{Istwert von C2}$
- ② M1: EIN, wenn $K100 = \text{Istwert von C2}$
- ③ M2: EIN, wenn $K100 < \text{Istwert von C2}$

M0, M1, und M2 bleiben unverändert, wenn die Eingangsbedingung X0 ausgeschaltet ist.

△

6.3.2 Numerische Datenbereiche vergleichen (ZCP, DZCP)

		ZCP		FNC 11			
		Numerische Datenbereiche vergleichen					
		CPU	FX1S	FX1N	FX2N		
			●	●	●		
Operanden	S1+, S2+, S3+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	Y, M, S	●	16 Bit	32 Bit	ZCP/ZCPP	9
				●	●	DZCP/DZCPP	17

Funktionsweise

Vergleich eines numerischen Datenwertes mit einem numerischen Datenbereich (größer, kleiner, gleich)

Beschreibung

- Die Daten in der Quelle (S3+) werden mit den Daten in den beiden Quellen (S1+) und (S2+) verglichen.
- Das Ergebnis des Vergleichs (größer, kleiner, gleich) wird durch Setzen eines Merkers M, Schrittstatusoperanden S oder Ausgangs Y angezeigt. Die Festlegung, welcher Operand gesetzt werden soll, erfolgt im Datenregister (D+).

$(S1+) > (S3+) \rightarrow (D+)$

$(S1+) < (S3+) < (S2+) \rightarrow ((D+)+1)$

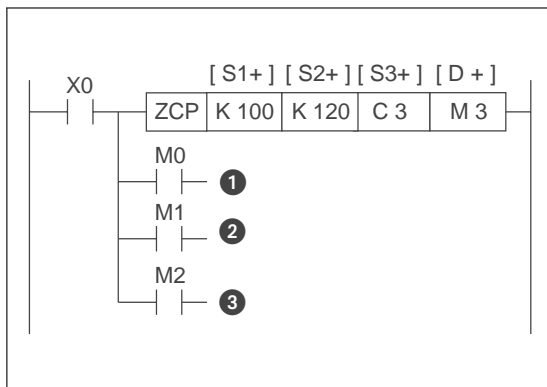
$(S2+) < (S3+) \rightarrow ((D+)+2)$

- Die Daten in (S1+) dürfen nicht größer sein als die Daten in (S2+).

Beispiel ▾

Steht in (S1+) der Wert „K100“ und in (S2+) der Wert „K90“, wird bei der Ausführung der ZCP-Anweisung davon ausgegangen, daß sich in (S2+) ebenfalls der Wert „K100“ befindet.

△

Beispiel ▾ Einsatz der ZCP-Anweisung**Abb. 6-19:**

Programmierbeispiel zum Einsatz der ZCP-Anweisung

C000072C

In der Zieladresse (D+) ist im Beispiel der Merker M3 angegeben. Entsprechend dem Ergebnis des Vergleichs werden demnach die Merker M3, M4 und M5 wie folgt geschaltet:

- ① M3: EIN, wenn $K100 > \text{Istwert von C3}$
- ② M4: EIN, wenn $K100 \leq \text{Istwert von C3} \leq K120$
- ③ M5: EIN, wenn $\text{Istwert von C3} > K120$

M3, M4 und M5 bleiben unverändert, wenn die Eingangsbedingung X0 ausgeschaltet ist.

Wenn sich der Istwert des Zählers C3 im Bereich von 100 bis 120 befindet, wird der Merker M4 eingeschaltet.

△

6.3.3 Datentransfer (MOV, DMOV)

		MOV		FNC 12		
		Datentransfer				
Operanden		S+		D+		
		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		KnY, KnM, KnS, T, C, D, V, Z		
		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
		●	16 Bit	32 Bit	MOV/MOVP	5
			●	●	DMOV/DMOVP	9

Funktionsweise

Übertragung von Daten aus einer Datenquelle zu einem Datenziel

Beschreibung

- Die Anweisung dient zur Übertragung von Daten aus einer Datenquelle (S+) zu einem Datenziel (D+).
- Die Daten in der Datenquelle (S+) werden automatisch bei der Ausführung der MOV-Anweisung als binärer Wert interpretiert.

HINWEIS

Die Anweisungen werden in jedem Programmzyklus ausgeführt. Dies können Sie durch Einsatz einer vorgeschalteten Impulsfunktion (PLS- oder PLF-Anweisung bzw. Parameter P) verhindern.

Beispiel ▾

Einsatz der MOV-Anweisung

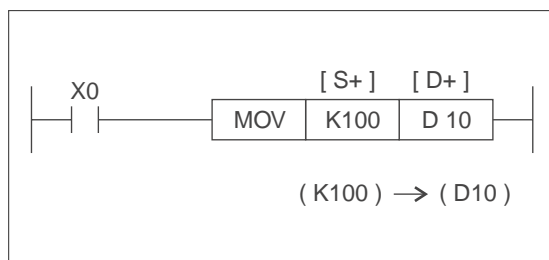


Abb. 6-20:
Programmierbeispiel zum Einsatz der MOV-Anweisung

C000073C

Ist die Eingangsbedingung X0 eingeschaltet, findet eine Übertragung der Daten von (S+) nach (D+) statt. Ist X0 ausgeschaltet, findet kein Datentransfer statt.

Die Konstante K100 wird bei der Ausführung der MOV-Anweisung automatisch als binärer Wert interpretiert. △

6.3.4 Shift-Transfer (SMOV)

				SMOV		FNC 13	
				Shift-Transfer			
				CPU	FX1S	FX1N	FX2N
				●			●
Operanden	S+	D+	n, m1, m2	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z 0 – 9999	K, H, KnY, KnM, KnS, T, C, D, V, Z	K, H 0 – 4	●	16 Bit	32 Bit	SMOV/ SMOVP
					●		11

Funktionsweise

Transfer von Daten und Verändern der Wertigkeit

Beschreibung

- Die Anweisung wird in 5 Schritten abgearbeitet:
 - 1.) Lesen der binären Daten aus (S+)
 - 2.) Wandeln der Daten in das BCD-Format
 - 3.) Verschieben der BCD-Stellen
 - 4.) Wandeln der Daten in das BIN-Format
 - 5.) Schreiben der binären Daten nach (D+)
- n, m1, m2 legen die Art der Verschiebung der BCD-Stellen fest.
 - m1 = 1. Stelle, die verschoben werden soll
 - m2 = Anzahl der Stellen, die verschoben werden sollen
 - n = 1. Zieladresse

Beispiel ▾

SMOV-Anweisung ohne Sondermerker

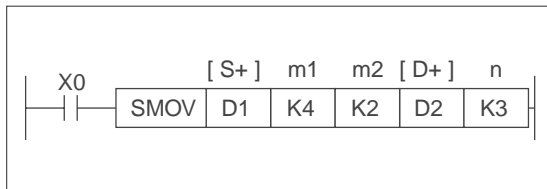


Abb. 6-21

Programmierbeispiel zum Einsatz der SMOV-Anweisung ohne Sondermerker

C000125C

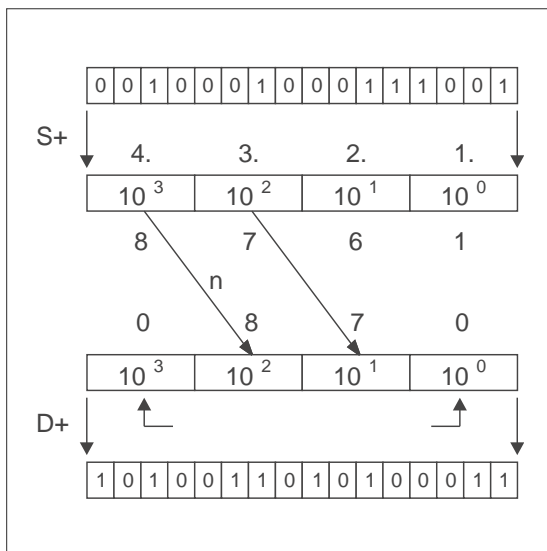


Abb. 6-22:

Konvertierung und Transfer

C000128C

Beispiel ▾

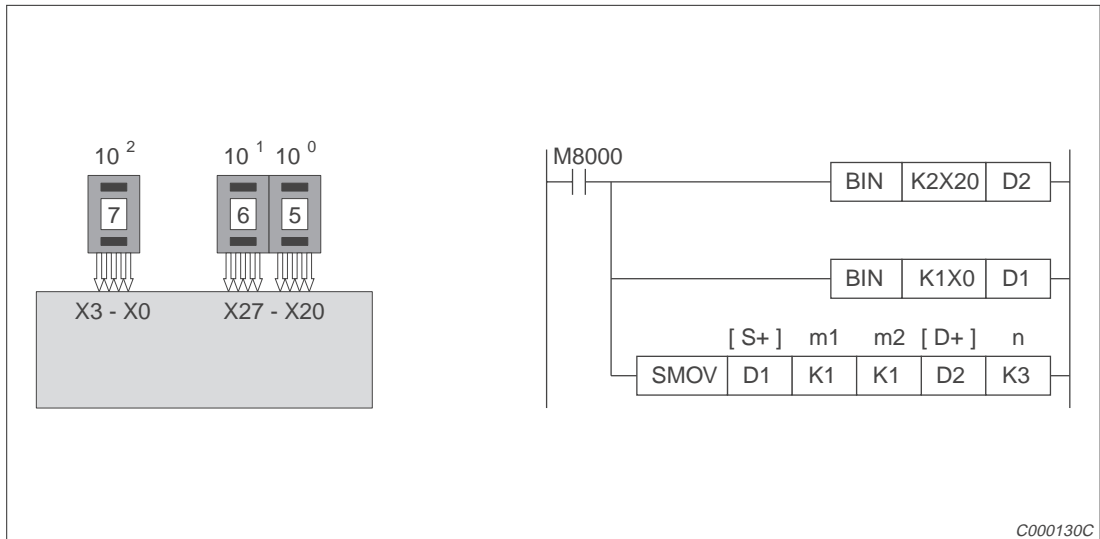


Abb. 6-23: Programmierbeispiel für Binäreingaben

Die Eingangsdaten werden von drei BCD-Schaltern geliefert, die entsprechenden Eingängen der Steuerung zugeordnet sind.

Zunächst werden die BCD-Daten der Eingänge X20 bis X27 (K2X20) in binäre Daten konvertiert und in D2 abgelegt.

Ebenso werden die Daten der Eingänge X0 bis X3 (K1X0) in binäre Daten umgewandelt. Das Ergebnis wird in D1 abgelegt.

Der BCD-Wert des Datenregisters D1 wird an die dritte Position im Zielregister D2 geschrieben. Anschließend werden die BCD-Daten wieder in binäre Daten umgewandelt.

In dem Beispiel werden die numerischen Eingangsdaten der drei BCD-Schalter zusammengefasst und im Datenregister D2 als binäre Daten gespeichert.



Funktionsweise mit Sondermerker M8168

Transfer von Daten im Hexadezimal-Format und Verändern der Wertigkeit

Beschreibung

- Die SMOV-Anweisung muß mit einer OUT-Anweisung kombiniert werden.
- Die Anweisung wird in 3 Schritten abgearbeitet:
 - 1.) Lesen der hexadezimalen Daten aus (S+), max. 4 Stellen, max. FFFF_H
 - 2.) Verschieben der Stellen
 - 3.) Schreiben der Daten nach (D+)
- n, m1, m2 legen die Art der Verschiebung der Stellen fest.
 - m1 = 1. Stelle, die verschoben werden soll
 - m2 = Anzahl der Stellen, die verschoben werden sollen
 - n = 1. Zieladresse

Beispiel ▾

Einsatz der SMOV-Anweisung mit Sondermerker M8168

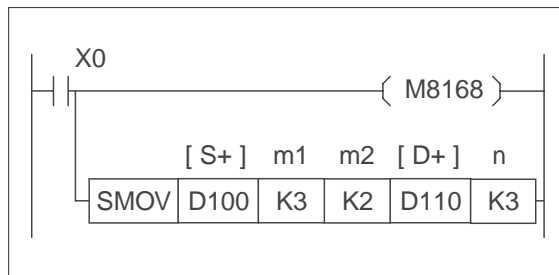


Abb. 6-24:
 Programmierbeispiel zum Einsatz der SMOV-Anweisung

C000312C

In Datenregister D100 ist die Zahl FFE2_H abgelegt, und im Zielregister D110 ist die Zahl 2CD9_H abgelegt. Wird X0 gesetzt, wird die SMOV-Anweisung ausgeführt.

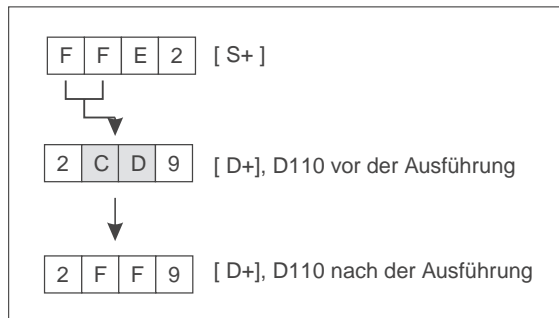


Abb. 6-25:
 Verschiebung in den Datenregistern

C000313C

Die Ziffern FF_H, 1. Stelle Nr. 4, 2 Stellen, von dem Wert in D100 werden kopiert und nach D110 verschoben. Dadurch werden die Ziffern CD_H, 1. Zieladresse Nr. 3, mit den Ziffern FF_H überschrieben.

△

6.3.5 Kopieren und invertieren (CML)

		CML		FNC 14						
		Kopieren und invertieren								
Operanden		S+ K, H, KnX, KnY, KnM, KnS, T, C, D, V Z		D+ KnY, KnM, KnS, T, C, D, V, Z		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
							●	16 Bit	32 Bit	CML, CMLP
						●		●	DCML, DCMLP	9

Funktionsweise

Bilden des 1er-Komplements einer binären Zahl

Beschreibung

Der binäre Zahlenwert in (S+) wird in sein 1er-Komplement gewandelt und nach (D+) geschrieben.

HINWEIS

Wenn die Zieladresse über eine größere Anzahl von Bits verfügt als die Quelladresse, werden alle nicht genutzten Bits eingeschaltet.

Beispiel ▾

CML-Anweisung

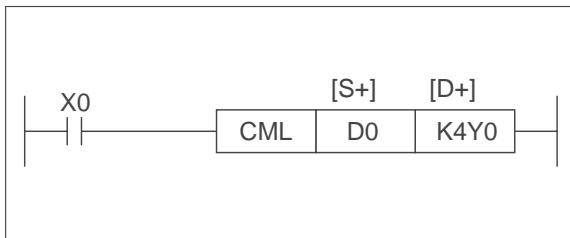
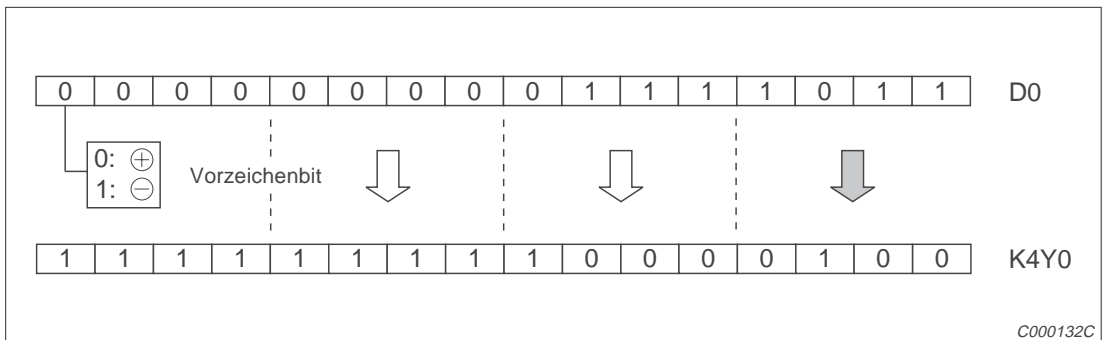


Abb. 6-26: Programmierbeispiel zur CML-Anweisung

C000131C



C000132C

Abb. 6-27: Invertierung und Transfer



6.3.6 Block-Transfer (BMOV)

				BMOV		FNC 15		
				Block-Transfer				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX,KnY,KnM,KnS, T,C,D,V,Z, File-Register	KnY,KnM,KnS, T,C,D,V,Z, File-Registe	K, H		●	16 Bit	32 Bit	BMOV/ BMOV P

Funktionsweise

Transferieren von Datenpaketen

Beschreibung

- Es wird eine vorgegebene Menge von Wortoperanden transferiert.
- Für den Transfer werden die Startadresse (S+), die Zieladresse (D+) und die Anzahl der zu transferierenden Worte (n) vorgegeben.
- Mit der BMOV-Anweisung können File-Register der MELSEC FX-Familie gelesen und geschrieben werden.

HINWEIS | Wenn die Größe des Datenpakets die Größe des Ziel- oder des Quellbereichs überschreitet, werden nur die in den Bereich passenden Worte übertragen.

Beispiel ▾ BMOV-Anweisung

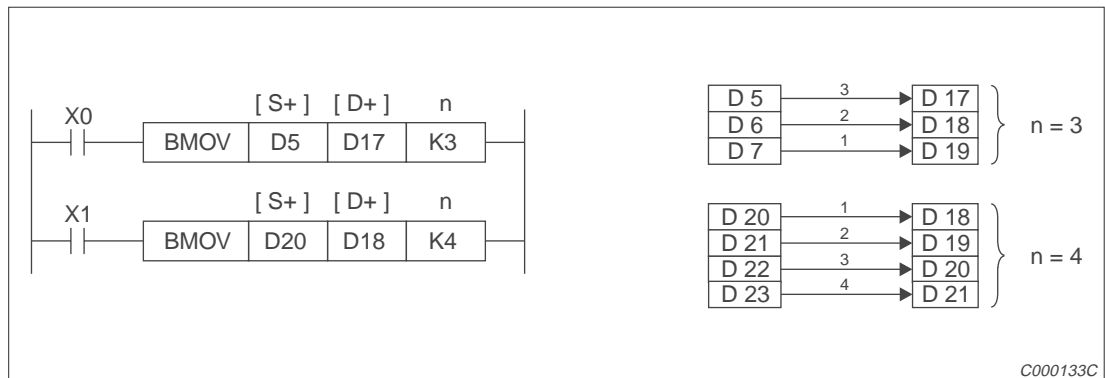


Abb. 6-28: Block-Transfer



6.3.7 Transfer von gleichen Daten (FMOV)

				FMOV		FNC 16		
				Transfer von gleichen Daten				
				CPU	FX1S	FX1N	FX2N	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX,KnY,KnM,KnS, T,C,D,V,Z,	KnY,KnM,KnS, T,C,D,V,Z	K, H		●	16 Bit	32 Bit	FMOV/ FMOV P
					●	●	DFMOV/ DFMOV P	13

Funktionsweise

Übertragen eines Datenwertes in mehrere Zieloperanden

Beschreibung

- Der Dateninhalt von (S+) wird in mehrere Zieloperanden gleichen Typs übertragen.
- Der erste Zieloperand wird in (D+) festgelegt.
- Ausgehend vom Zieloperanden (D+) wird der Datenwert aus (S+) in n Operanden übertragen.

HINWEIS

Wenn n größer als die Anzahl verfügbarer Operanden ist, erfolgt die Übertragung nur bis zum letzten verfügbaren Operanden.

Beispiel ▾

FMOV-Anweisung

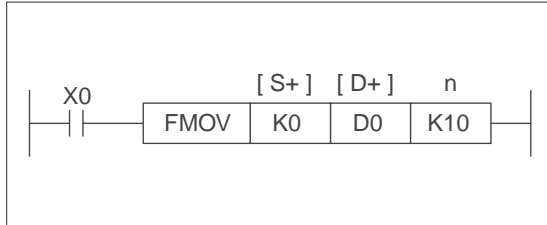


Abb. 6-29:
Programmierbeispiel zur FMOV-Anweisung

C000134C

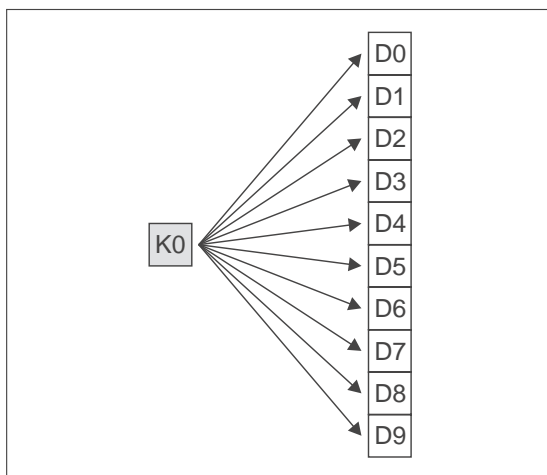


Abb. 6-30:
Datentransfer des Wertes „0“ in die Datenregister D0 – D9

C000119C



6.3.8 Austausch von Daten (XCH)

		XCH		FNC 17			
		Austausch von Daten					
Operanden		D1+, D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				CPU	FX1S		FX1N
		KnY, KnM, KnS T, C, D, V, Z	●	16 Bit	32 Bit	XCH/XCHP	5
				●	●	DXCH/DXCHP	9

Funktionsweise

Austausch von Daten zwischen zwei Operanden

Beschreibung

Die Daten von (D1+) und (D2+) werden ausgetauscht.

HINWEIS

Der Austauschvorgang wird in jedem Zyklus ausgeführt, wenn keine Flankensteuerung programmiert wird.

Beispiel ▾

XCH-Anweisung ohne Sondermerker M8160

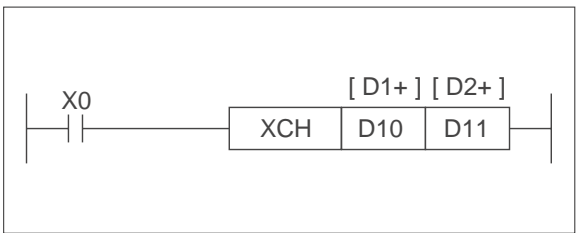


Abb. 6-31:
Programmierbeispiel zur XCH-Anweisung

Werte vor der Ausführung: D10 = 5, D11 = 7

Werte nach der Ausführung: D10 = 7, D11 = 5

C000135C



Funktionsweise mit Sondermerker M8160

Bei gesetztem Sondermerker M8160 erfolgt ein Austausch der oberen und unteren Bytes in (D1+) und (D2+).

Beschreibung

Nach Setzen des Sondermerkers M8160 werden in (D1+) und in (D2+) das obere Byte und das untere Byte ausgetauscht. Wird die XCH-Anweisung ohne Sondermerker M8160 noch einmal in dem Programm verwendet, muß der Sondermerker M8160 wieder zurückgesetzt werden.

HINWEIS

Bei Verwendung der XCH-Anweisung mit Sondermerker M8160 müssen (D1+) und (D2+) das gleiche Datenregister angeben, andernfalls kommt es zu einer Fehlermeldung, Fehler-Flag M8067.

Der Austauschvorgang wird in jedem Zyklus ausgeführt, wenn keine Flankensteuerung programmiert wird.

Beispiel ▽

DXCH-Anweisung mit Sondermerker M8160

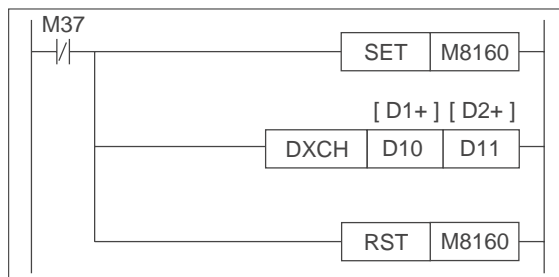


Abb. 6-32:
 Programmierbeispiel zur
 DXCH-Anweisung mit Sondermerker
 M8160

C000314C

Der Austauschvorgang läßt sich wie folgt darstellen:

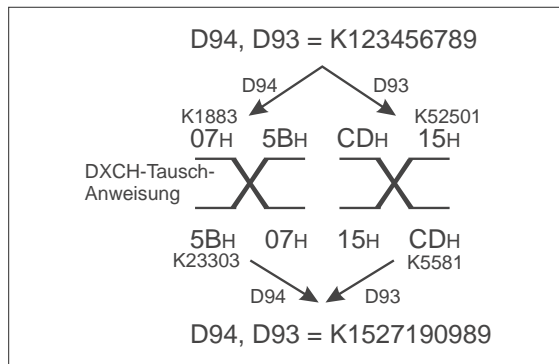


Abb. 6-33:
 Darstellung des Austauschvorgangs

C000315C



HINWEIS

Bei Anwendung der 32-Bit-Anweisung DXCH/DXCHP erfolgt der Austausch des oberen und unteren Bytes unabhängig in jedem Einzelwort (16 Bit).

6.3.9 BCD-Konvertierung (BCD, DBCD)

			BCD		FNC 18		
			BCD-Konvertierung				
			CPU	FX1S	FX1N	FX2N	
				●	●	●	
Operanden	S+	D+	Puls-Anweisung (P) ●	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		16 Bit	32 Bit	BCD/BCDP	5
				●	●	DBCD/DBCDP	9

Funktionsweise

Konvertierung von binären Daten in ein BCD-Format

Beschreibung

Innerhalb der Steuerung werden nur binäre Daten verarbeitet. Durch Einsatz der BCD-Anweisung können auch Daten in einem BCD-Format ausgegeben werden (z. B. zum Ansteuern einer 7-Segment-Anzeige).

- Die binären Daten in der Quelle (S+) werden in BCD-Daten konvertiert und zur Zieladresse (D+) übertragen.
- Das Ergebnis der BCD-Konvertierung muß innerhalb des zulässigen Bereiches liegen:
 16-Bit-Anweisung: 0 bis +9 999
 32-Bit-Anweisung: 0 bis +99 999 999

Fehlerquelle

Liegt das Ergebnis der BCD-Konvertierung außerhalb des zulässigen Bereichs, tritt ein Programmablauffehler auf, und die Anweisung wird nicht ausgeführt.

Beispiel ▾

Die BCD-Anweisung können Sie beispielsweise einsetzen, um binäre Daten aus der SPS zu lesen und auf einer 7-Segment-Anzeige darzustellen.

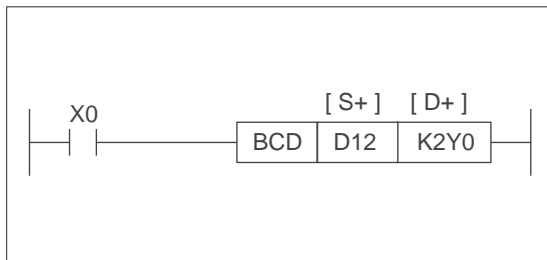
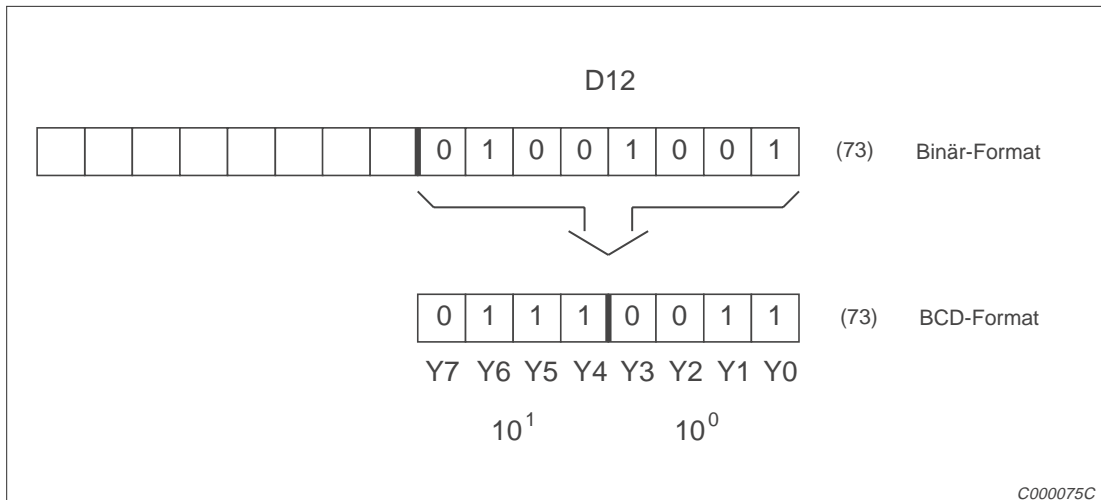


Abb. 6-34:
 Programmierbeispiel zum Einsatz der BCD-Anweisung

C000074C



C000075C

Abb. 6-35: Programmierbeispiel zur Konvertierung von binären Daten in ein BCD-Format

Die binären Daten aus dem Datenregister D12 werden in ein BCD-Format konvertiert und danach über die Ausgänge Y0 bis Y7 ausgegeben. In diesem Beispiel: 73 (dezimal). ▴

6.3.10 Binär-Konvertierung (BIN, DBIN)

		BIN		FNC 19			
		Binär-Konvertierung					
		CPU	FX1S	FX1N	FX2N		
		●	●	●	●		
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	BIN/BINP	5
				●	●	DBIN/DBINP	9

Funktionsweise

Konvertierung von BCD-Daten in ein binäres Format

Beschreibung

Innerhalb der Steuerung werden nur binäre Daten verarbeitet. Durch Einsatz der BIN-Anweisung können auch Daten in einem BCD-Format über die Eingänge eingelesen werden.

- Die BCD-Daten in der Quelle (S+) werden in binäre Daten konvertiert und zur Zieladresse (D+) übertragen.
- Die Daten in (S+) müssen innerhalb des zulässigen Bereiches liegen:
 16-Bit-Anweisung: 0 bis +9 999
 32-Bit-Anweisung: 0 bis +99 999 999

Fehlerquelle

Sind die Daten in (S+) nicht im BCD-Format, tritt ein Fehler auf. Der Fehler wird durch den eingeschalteten Sondermerker M8067 angezeigt. Der Sondermerker M8068 wird nicht eingeschaltet (siehe auch Abs. 11.1.1).

Beispiel ▾

Die BIN-Anweisung können Sie z. B. einsetzen, um BCD-Daten von digitalen Schaltern in die SPS zu übertragen.

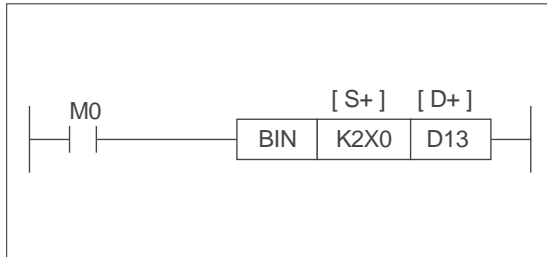
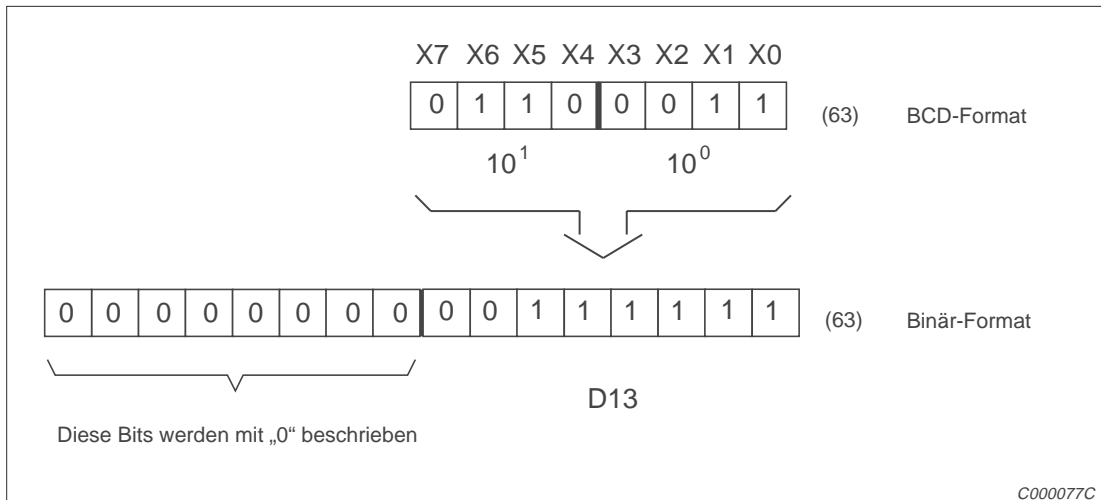


Abb. 6-36:
 Programmierbeispiel für den Einsatz der BIN-Anweisung

C000076C



C000077C

Abb. 6-37: Programmierbeispiel für die Konvertierung von Daten im BCD-Format in ein binäres Datenformat

Die BCD-Daten an den Eingängen X0 bis X7 werden in ein binäres Datenformat konvertiert. Anschließend werden die Daten zur Zieladresse D13 übertragen. △

6.4 Arithmetische Anweisungen

Übersicht der Anweisungen FNC 20 bis 29

Symbol	FNC	Bedeutung	Abschnitt
ADD	20	Addition numerischer Daten	6.4.1
SUB	21	Subtraktion numerischer Daten	6.4.2
MUL	22	Multiplikation numerischer Daten	6.4.3
DIV	23	Division numerischer Daten	6.4.4
INC	24	Inkrementieren	6.4.5
DEC	25	Dekrementieren	6.4.6
WAND	26	logische UND-Verknüpfung	6.4.7
WOR	27	logische ODER-Verknüpfung	6.4.8
WXOR	28	logische Exklusiv-ODER-Verknüpfung	6.4.9
NEG	29	Negation von Daten	6.4.10

Tab. 6-11: Übersicht der Anweisungen FNC 20 bis 29

6.4.1 Addition numerischer Daten (ADD, DADD)

			ADD		FNC 20		
			Addition numerischer Daten				
			CPU	FX1S	FX1N	FX2N	
				●	●	●	
Operanden	S+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	ADD/ADDP	7
				●	●	DADD/DADDP	13

Funktionsweise

Addition von zwei numerischen Daten. Das Ergebnis der Addition wird in einer Zieladresse gespeichert.

Beschreibung

- Die binären Daten in den Quelladressen (S1+) und (S2+) werden addiert. Das Ergebnis der Addition wird in (D+) abgespeichert.
 $(S1+) + (S2+) = (D+)$
- In dem höchstwertigen Bit wird das Vorzeichen der Addition gespeichert:
0: positives Vorzeichen
1: negatives Vorzeichen
- Bei der Ausführung einer 32-Bit-Anweisung wird der Wortoperand der unteren 16 Bit in der Anweisung angegeben. Der darauffolgende Operand ist der Wortoperand der oberen 16 Bit. Es wird empfohlen, bei der Angabe der Adressen gerade Zahlen zu verwenden, damit nicht versehentlich überschneidende Adressen programmiert werden.
- In der Quelladresse (S+) und der Zieladresse (D+) kann auch der gleiche Operand angegeben werden.

HINWEISE

Bei bestimmten Rechenergebnissen wird nach Ausführung der Anweisung ein Sondermerker (Flag) gesetzt.

Zero Flag M8020

Lautet das Ergebnis der Addition 0, wird das Zero Flag gesetzt.

Borrow Flag M8021

Unterschreitet das Ergebnis der Addition -32 767 (16-Bit-Operation) bzw. -2 147 483 648 (32-Bit-Operation), wird das Borrow Flag gesetzt.

Carry Flag M8022

Überschreitet das Ergebnis den Wert +32 767 (16-Bit-Operation) bzw. +2 147 483 647 (32-Bit-Operation), wird das Carry Flag gesetzt.

Beispiel ▾

Einsatz der ADD-Anweisung

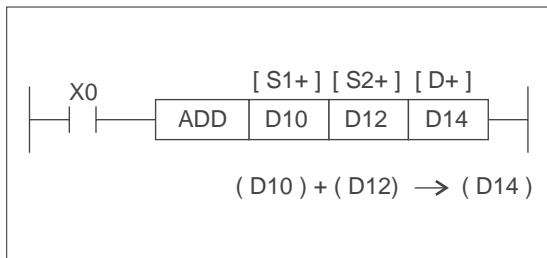


Abb. 6-38:

Programmierbeispiel zum Einsatz der ADD-Anweisung

C000078C

Ist X0 eingeschaltet, werden die Datenwerte in den Registern D10 und D12 addiert. Das Ergebnis der Addition wird im Datenregister D14 abgespeichert.

Einsatz der DADD-Anweisung

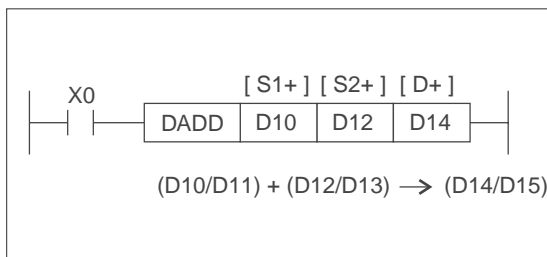


Abb. 6-39:

Programmierbeispiel zum Einsatz der DADD-Anweisung

C000069C



6.4.2 Subtraktion numerischer Daten (SUB, DSUB)

			SUB		FNC 21		
			Subtraktion numerischer Daten				
			CPU	FX1S	FX1N	FX2N	
			●	●	●	●	
Operanden	S+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	SUB/SUBP	7
				●	●	DSUB/DSUBP	13

Funktionsweise

Subtraktion zweier numerischer Daten. Das Ergebnis der Subtraktion wird in einer Zieladresse abgespeichert.

Beschreibung

- Der Datenwert in (S2+) wird von dem Datenwert in (S1+) subtrahiert. Das Ergebnis der Subtraktion wird in (D+) abgelegt.
 $(S1+) - (S2+) = (D+)$
- In dem höchstwertigen Bit wird das Vorzeichen der Addition gespeichert:
 0: positives Vorzeichen
 1: negatives Vorzeichen
- Bei der Ausführung einer 32-Bit-Anweisung wird der Wortoperand der unteren 16 Bit in der Anweisung angegeben. Der darauffolgende Operand ist der Wortoperand der oberen 16 Bit. Es wird empfohlen, bei der Angabe der Adressen gerade Zahlen zu verwenden, damit nicht versehentlich dieselben Adressen programmiert werden.
- In der Quelladresse (S+) und der Zieladresse (D+) kann auch der gleiche Operand angegeben werden.

HINWEISE

Bei bestimmten Rechenergebnissen wird nach Ausführung der Anweisung ein Sondermerker (Flag) gesetzt.

Zero Flag M8020

Lautet das Ergebnis der Subtraktion 0, wird das Zero Flag gesetzt.

Borrow Flag M8021

Unterschreitet das Ergebnis der Subtraktion -32 767 (16-Bit-Operation) bzw. -2 147 483 648 (32-Bit-Operation), wird das Borrow Flag gesetzt.

Carry Flag M8022

Überschreitet das Ergebnis den Wert +32 767 (16-Bit-Operationen) bzw. +2 147 483 647 (32-Bit-Operationen), wird das Carry Flag gesetzt.

Beispiel ▾

Einsatz der SUB-Anweisung

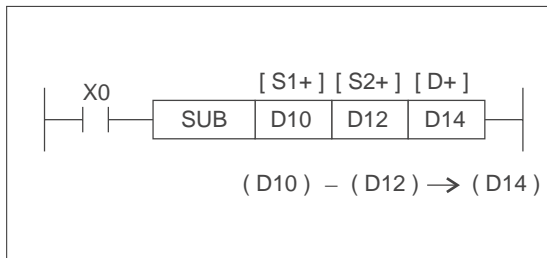


Abb. 6-40:

Programmierbeispiel zum Einsatz der SUB-Anweisung

C000079C

Ist X0 eingeschaltet, wird der Datenwert im Datenregister D12 von dem Datenwert im Datenregister D10 subtrahiert. Das Ergebnis der Subtraktion wird im Datenregister D14 abgespeichert.

Einsatz der DSUB-Anweisung

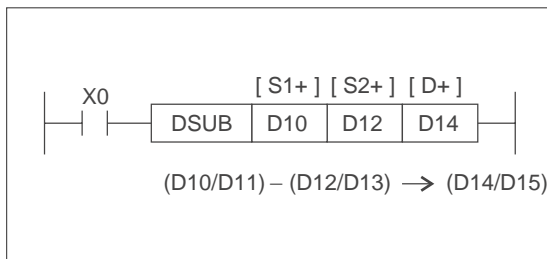


Abb. 6-41:

Programmierbeispiel zum Einsatz der DSUB-Anweisung

C000067C



6.4.3 Multiplikation numerischer Daten (MUL, DMUL)

			MUL		FNC 22		
			Multiplikation numerischer Daten				
			CPU	FX1S	FX1N	FX2N	
					●	●	●
Operanden	S+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, Z (V)		●	16 Bit	32 Bit	MUL/MULP
				●	●	DMUL/DMULP	13

Funktionsweise

Multiplikation von zwei numerischen Daten. Das Ergebnis der Multiplikation wird in einer Zieladresse abgespeichert.

Beschreibung

- Die Daten in S1+ und S2+ werden miteinander multipliziert. Das Ergebnis der Multiplikation wird in der in D+ angegebenen Operandenadresse und den darauffolgenden Operandenadressen abgespeichert.

$$(S1+) \times (S2+) = (D+)$$

- In dem höchstwertigen Bit wird das Vorzeichen des Multiplikationsergebnisses abgespeichert.

0: positives Vorzeichen
1: negatives Vorzeichen

- Bei der Ausführung einer 16-Bit-Operation wird das Ergebnis als 32-Bit-Zahl in (D+) und ((D+)+1) abgelegt. Bei der Ausführung einer 32-Bit-Operation wird das Ergebnis als 64-Bit-Zahl in (D+) und den drei darauffolgenden Operanden abgelegt.
- Bei der 32-Bit-Operation kann für (D+) nicht Z(V) verwendet werden.

Multiplikation von 16-Bit-Daten (MUL-Anweisung)

Das Ergebnis einer 16-Bit-Multiplikation ergibt eine 32-Bit-Zahl. Diese Zahl wird als 32-Bit-Datenwert abgespeichert. Die unteren 16 Bit werden in der in D+ angegebenen Operandenadresse abgespeichert. Die oberen 16 Bit werden in den darauffolgenden Operandenadressen abgespeichert.

Beispiel ▾

Einsatz der MUL-Anweisung

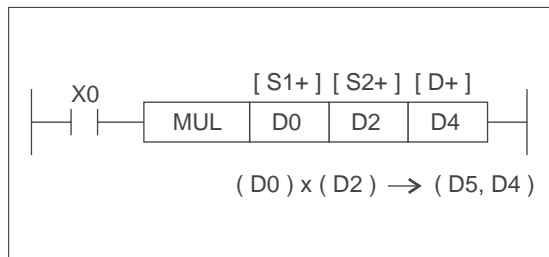


Abb. 6-42:

Programmierbeispiel zum Einsatz der MUL-Anweisung

C000080C

Das Ergebnis der Multiplikation wird als 32-Bit-Datenwert in den Datenregistern D4 und D5 abgelegt. In D4 stehen die unteren 16 Bit, und in D5 stehen die oberen 16 Bit. Das Vorzeichen des Multiplikationsergebnisses steht im 15. Bit von D5. △

Multiplikation von 32-Bit-Daten (DMUL-Anweisung)

Das Ergebnis einer 32-Bit-Multiplikation wird als 64-Bit-Datenwert abgespeichert. Die unteren 16 Bit werden in der in D+ angegebenen Operandenadresse abgespeichert. Die höherwertigen Bit werden in den darauffolgenden Operandenadressen abgespeichert.

Beispiel ▾

Einsatz der DMUL-Anweisung

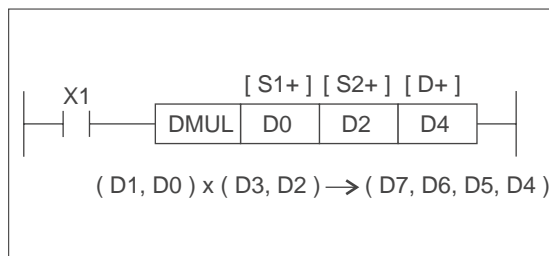


Abb. 6-43:

Programmierbeispiel zum Einsatz der DMUL-Anweisung

C000081C

Das Ergebnis der Multiplikation wird als 64-Bit-Datenwert in den Datenregistern D4, D5, D6 und D7 abgelegt. In D4 stehen die unteren 16 Bit, und in D5, D6 und D7 stehen die höherwertigen Bit. △

6.4.4 Division numerischer Daten (DIV, DDIV)

		DIV		FNC 23				
		Division numerischer Daten						
Operanden S1+, S2+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		D+ KnY, KnM, KnS, T, C, D, Z (V)		CPU	FX1S	FX1N	FX2N	
				●	●	●		
Operanden S1+, S2+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		D+ KnY, KnM, KnS, T, C, D, Z (V)		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				●	16 Bit	32 Bit	DIV/DIVP	7
					●	●	DDIV/DDIVP	13

Funktionsweise

Division zwischen zwei numerischen Daten. Das Ergebnis der ganzzahligen Division wird in einer Zieladresse abgespeichert.

Beschreibung

- Es findet eine Division zwischen den Daten in (S1+) und den Daten in (S2+) statt. Die Daten in (S1+) entsprechen dem Dividenten. Die Daten in (S2+) entsprechen dem Divisor. Das Ergebnis der Division wird in der in (D+) angegebenen Operandenadresse und in den darauffolgenden Operandenadressen abgespeichert. Der Teilungsrest wird in einer der darauffolgenden Operandenadressen abgespeichert.

$$(S1+) : (S2+) = (D+)$$

- In dem höchstwertigen Bit wird das Vorzeichen der Division abgespeichert.

0: positives Vorzeichen
 1: negatives Vorzeichen

Das Vorzeichen des Divisionsergebnisses ist abhängig von den Vorzeichen des Dividenten und des Divisors.

- Bei der 32-Bit-Operation kann für (D+) nicht Z(V) verwendet werden.

Divident	Divisor	Quotient	Teilungsrest
+	+	+	+
+	-	-	+
-	+	-	-
-	-	+	-

Tab. 6-12:
 Vorzeichen des Divisionsergebnisses

HINWEIS

Ist die Zieladresse ein Bit-Operand, kann die Steuerung keinen Teilungsrest ermitteln.

Fehlerquelle

Es kommt zu einem Programmablauffehler, wenn der Wert des Divisors gleich 0 ist. Die Anweisung wird nicht abgearbeitet.

Division von 16-Bit-Daten (DIV-Anweisung)

Das Ergebnis einer 16-Bit-Division wird in der in D+ angegebenen Operandenadresse abgespeichert. Der Teilungsrest wird in der darauffolgenden Operandenadresse abgespeichert.

Beispiel ▾ Einsatz der DIV-Anweisung

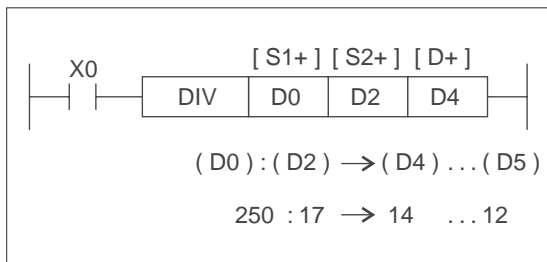


Abb. 6-44:
Programmierbeispiel zum Einsatz der DIV-Anweisung

C000082C

Das Ergebnis der Division 14 wird im Datenregister D4 gespeichert. Der Teilungsrest 12 wird im darauffolgenden Datenregister D5 abgelegt. △

Division von 32-Bit-Daten (DDIV-Anweisung)

Bei einer Division von 32-Bit-Daten stehen für den Dividenten, den Divisor, das Ergebnis und den Teilungsrest jeweils zwei aufeinanderfolgende Datenregister zur Verfügung. In der DDIV-Anweisung müssen Sie jeweils das Datenregister mit der niedrigeren Operandenadresse angeben.

Beispiel ▾ Einsatz der DDIV-Anweisung

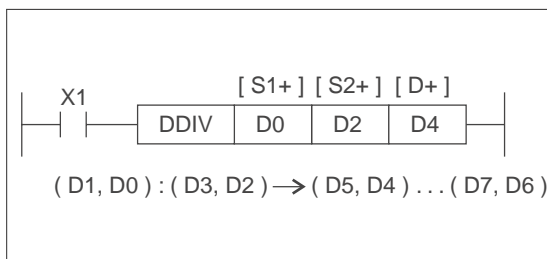


Abb. 6-45:
Programmierbeispiel zum Einsatz der DDIV-Anweisung

C000083C

Das Ergebnis der Division wird im Datenregister D4 und D5 abgespeichert. Der Teilungsrest wird in den darauffolgenden Datenregister D6 und D7 abgespeichert. △

6.4.5 Inkrementieren (INC, DINC)

		INC		FNC 24		
		Inkrementieren				
Operanden	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
			16 Bit	32 Bit	INC/INCP	3
	KnY, KnM, KnS, T, C, D, V, Z	●	●	●	DINC/DINCP	5

Funktionsweise

Zu einem numerischen Datenwert wird die Zahl 1 addiert (inkrementiert).

HINWEIS

Die Anweisung wird in jedem Programmzyklus ausgeführt. Dies können Sie durch Einsatz einer vorgeschalteten Impulsfunktion (PLS- oder PLF-Anweisung) oder den Einsatz des Befehlsparameters P (MELSEC FX und FX2N) verhindern.

Beim Inkrementieren wird zu dem in D+ gespeicherten Wert die Zahl 1 addiert, sobald die Eingangsbedingung erfüllt ist.

- **16-Bit-Operation (INC-Anweisung)**
Wird bei einer 16-Bit-Operation der Wert 1 zu dem Wert +32 767 addiert, lautet das Ergebnis -32 768. Es wird kein Flag gesetzt.
- **32-Bit-Operation (DINC-Anweisung)**
Wird bei einer 32-Bit-Operation der Wert 1 zu dem Wert +2 147 483 647 addiert, lautet das Ergebnis -2 147 483 648. Es wird kein Flag gesetzt.

Beispiel ▾

Einsatz der INC-Anweisung

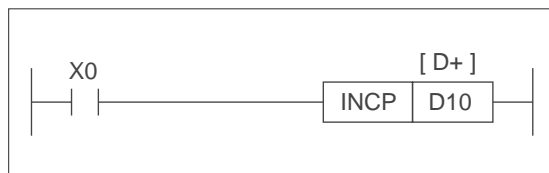


Abb. 6-46:
Programmierbeispiel zum Einsatz der INCP-Anweisung

C000136C

Der Datenwert im Datenregister D10 wird bei jedem Anliegen eines Eingangssignals X0 um den Zahlenwert 1 erhöht.

Die Anweisung wird durch eine vorgeschaltete Impulsfunktion aktiviert. Dies ist wichtig, damit der Additionsvorgang nicht in jedem Programmzyklus stattfindet. ▴

6.4.6 Dekrementieren (DEC)

		DEC		FNC 25			
		Dekrementieren					
Operanden		D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				16 Bit	32 Bit	DEC/DECP	3
		KnY, KnM, KnS, T, C, D, V, Z	●	●	●	DDEC/DDECP	5

Funktionsweise

Von einem numerischen Datenwert wird die Zahl 1 subtrahiert (dekrementiert).

HINWEIS

Die Anweisung wird in jedem Programmzyklus ausgeführt. Dies können Sie durch Einsatz einer vorgeschalteten Impulsfunktion (PLS- oder PLF-Anweisung) oder den Einsatz des Befehlsparameters P (MELSEC FX/FX2N) verhindern.

Dekrementieren

Beim Dekrementieren wird die Zahl 1 von dem in D+ gespeicherten Wert subtrahiert, sobald die Eingangsbedingung erfüllt ist.

- **16-Bit-Operation (DEC-Anweisung)**
Wird bei einer 16-Bit-Operation der Wert 1 von dem Wert -32 768 subtrahiert, lautet das Ergebnis +32 767. Es wird kein Flag gesetzt.
- **32-Bit-Operation (DDEC-Anweisung)**
Wird bei einer 32-Bit-Operation der Wert 1 von dem Wert -2 147 483 648 subtrahiert, lautet das Ergebnis +2 147 483 647. Es wird kein Flag gesetzt.

Beispiel ▾

Einsatz der DEC-Anweisung

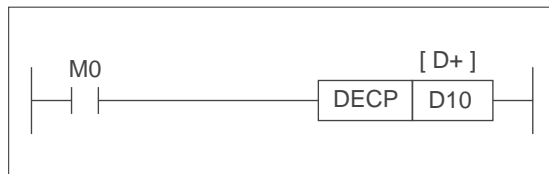


Abb. 6-47:
Programmierbeispiel zum Einsatz der DECP-Anweisung

C000066C

Der Datenwert im Datenregister D10 wird bei jeder Betätigung von X0 um den Zahlenwert 1 vermindert.

Die Anweisung wird durch eine vorgeschaltete Impulsfunktion aktiviert. Dies ist wichtig, damit der Subtraktionsvorgang nicht in jedem Programmzyklus stattfindet. △

6.4.7 Logische UND-Verknüpfung binärer Daten (WAND, DAND)

		WAND		FNC 26		
		Logische UND-Verknüpfung				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	WAND/ WANDP

		DAND		FNC 26		
		Logische UND-Verknüpfung				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	DAND/ DANDP

Funktionsweise

Logische UND-Verknüpfung binärer Daten

Beschreibung

- Es wird eine logische UND-Verknüpfung von einzelnen Bits durchgeführt.
- Die Daten in (S1+) und (S2+) werden bitweise miteinander verknüpft. Das Verknüpfungsergebnis wird in (D+) abgespeichert.

(S1+)	(S2+)	(D+)
1	1	1
1	0	0
0	1	0
0	0	0

Tab. 6-13:
Wahrheitstabelle der UND-Verknüpfung

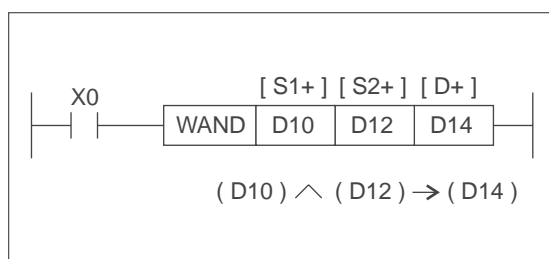


Abb. 6-48:
Programmierbeispiel zum Einsatz der WAND-Anweisung

C000086C

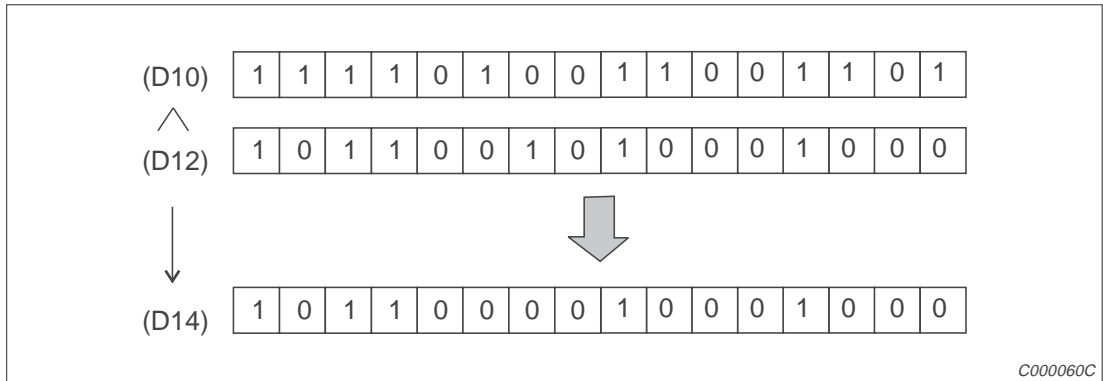


Abb. 6-49: Einsatz der WAND-Anweisung

6.4.8 Logische ODER-Verknüpfung binärer Daten (WOR, DOR)

			WOR		FNC 27																						
			Logische ODER-Verknüpfung																								
<table border="1"> <tr> <th>Operanden</th> <th>S1+, S2+</th> <th>D+</th> <th>Puls-Anweisung (P)</th> <th colspan="2">Verarbeitung</th> <th>Programmschritte</th> </tr> <tr> <td></td> <td>K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z</td> <td>KnY, KnM, KnS, T, C, D, V, Z</td> <td>●</td> <td>16 Bit</td> <td>32 Bit</td> <td>WOR/ WORP</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td>●</td> <td></td> <td>7</td> </tr> </table>			Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	WOR/ WORP					●		7	CPU	FX1S	FX1N	FX2N
			Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte																		
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	WOR/ WORP																					
				●		7																					
				●																							

			DOR		FNC 27																						
			Logische ODER-Verknüpfung																								
<table border="1"> <tr> <th>Operanden</th> <th>S1+, S2+</th> <th>D+</th> <th>Puls-Anweisung (P)</th> <th colspan="2">Verarbeitung</th> <th>Programmschritte</th> </tr> <tr> <td></td> <td>K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z</td> <td>KnY, KnM, KnS, T, C, D, V, Z</td> <td>●</td> <td>16 Bit</td> <td>32 Bit</td> <td>DOR/ DORP</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> <td>●</td> <td>13</td> </tr> </table>			Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	DOR/ DORP						●	13	CPU	FX1S	FX1N	FX2N
			Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte																		
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	DOR/ DORP																					
					●	13																					

Funktionsweise

Logische ODER-Verknüpfung binärer Daten

Beschreibung

- Es wird eine logische ODER-Verknüpfung von einzelnen Bits durchgeführt.
- Die Daten in (S1+) und (S2+) werden bitweise miteinander verknüpft. Das Verknüpfungsergebnis wird in (D+) abgespeichert.

(S1+)	(S2 +)	(D+)
1	1	1
1	0	1
0	1	1
0	0	0

Tab. 6-14:
Wahrheitstabelle der ODER-Verknüpfung

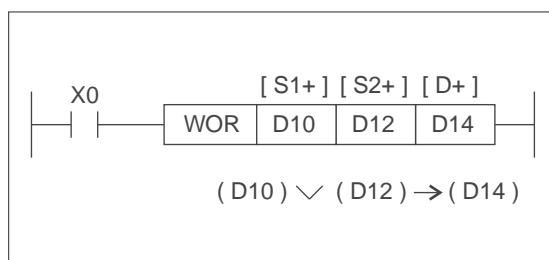


Abb. 6-50:
Programmierbeispiel zum Einsatz der WOR-Anweisung

C000087C

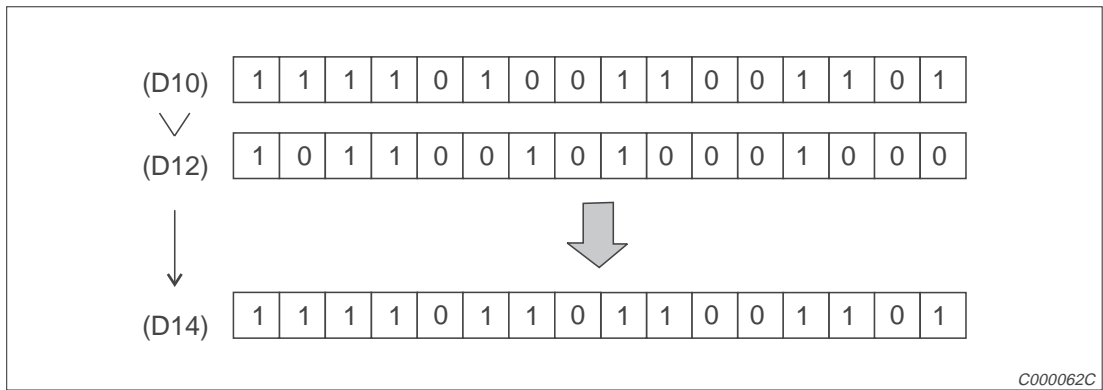


Abb. 6-51: Einsatz der WOR-Anweisung

6.4.9 Logische Exklusiv-ODER-Verknüpfung binärer Daten (WXOR, DXOR)

		WXOR		FNC 28						
		Logische Exklusiv-ODER-Verknüpfung								
Operanden		S1+, S2+		D+		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		KnY, KnM, KnS, T, C, D, Z (V)			●	16 Bit	32 Bit	WXOR/ WXORP
						●		●	●	

		DXOR		FNC 29						
		Logische Exklusiv-ODER-Verknüpfung								
Operanden		S1+, S2+		D+		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		KnY, KnM, KnS, T, C, D, Z (V)			●	16 Bit	32 Bit	DXOR/ DXORP
						●		●	●	

Funktionsweise

Logische Exklusiv-ODER-Verknüpfung binärer Daten

Beschreibung

- Es wird eine logische Exklusiv-ODER-Verknüpfung von einzelnen Bits durchgeführt.
- Die Daten in (S1+) und (S2+) werden bitweise miteinander verknüpft. Das Verknüpfungsergebnis wird in (D+) abgespeichert.

(S1+)	(S2+)	(D+)
1	1	0
1	0	1
0	1	1
0	0	0

Tab. 6-15:

Wahrheitstabelle der Exklusiv-ODER-Verknüpfung

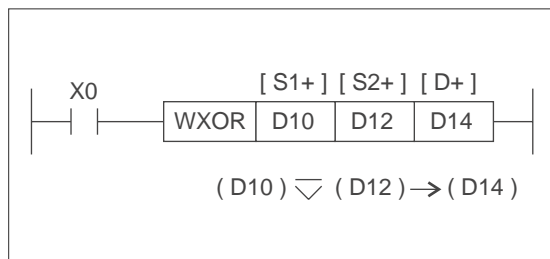


Abb. 6-52:

Programmierbeispiel zum Einsatz der WXOR-Anweisung

C000088C

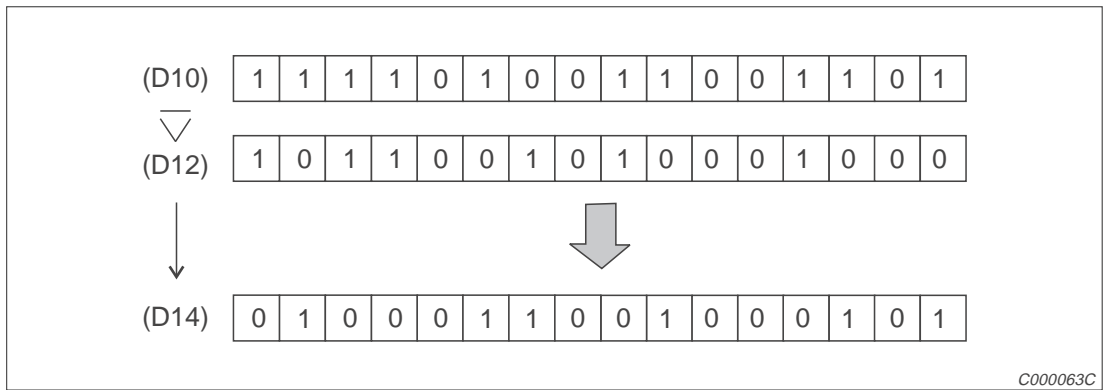


Abb. 6-53: Beispiel zum Einsatz der WXOR-Anweisung

6.4.10 Negation von Daten (NEG)

		NEG		FNC 29		
		Negation von Daten				
Operanden	D KnY, KnM, KnS T, C, D, V, Z	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
			CPU	FX1S		FX1N
		●	●	●	●	
			16 Bit	32 Bit	NEG/NEGP	3
			●	●	DNEG/DNEGP	5

Funktionsweise

Bilden des 2er-Komplements eines Datenwertes

Beschreibung

- Durch die Anweisung NEG wird das 2er-Komplement des in (D+) angegebenen Datenwertes gebildet und in (D+) abgespeichert.

HINWEIS

Wenn keine Flankenerkennung programmiert wird, wird die Komplementbildung in jedem Zyklus wiederholt.

Beispiel ▾

NEG-Anweisung

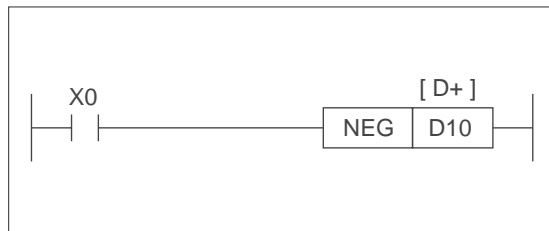


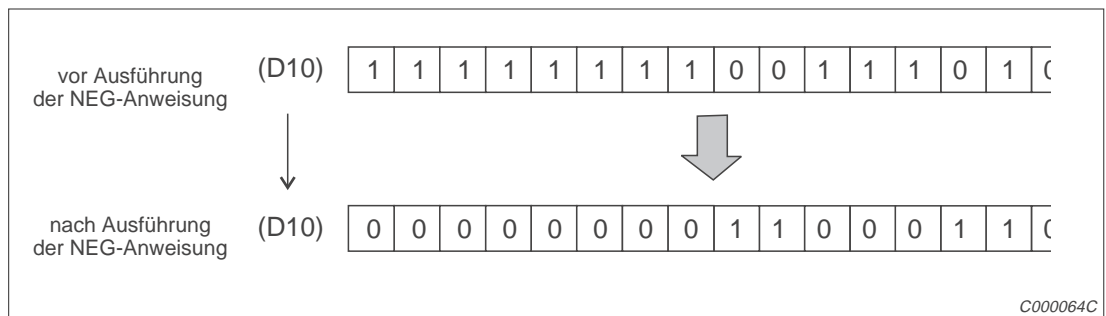
Abb. 6-54

Programmierbeispiel zur NEG-Anweisung

C000137C

Funktion

binär: $\overline{D10} + 1 \rightarrow D10$



C000064C

Abb. 6-55: Funktion der NEG-Anweisung



6.5 Verschiebeanweisungen

Übersicht der Anweisungen FNC 30 bis 39

Symbol	FNC	Bedeutung	Abschnitt
ROR	30	Rotation nach rechts	6.5.1
ROL	31	Rotation nach links	6.5.2
RCR	32	Rotieren von Bits nach rechts	6.5.3
RCL	33	Rotieren von Bits nach links	6.5.4
SFTR	34	Binäre Daten bitweise verschieben, rechts	6.5.5
SFTL	35	Binäre Daten bitweise verschieben, links	6.5.5
WSFR	36	Daten wortweise nach rechts verschieben	6.5.6
WSFL	37	Daten wortweise nach links verschieben	6.5.7
SFWR	38	Schreiben in einen FIFO-Speicher	6.5.8
SFRD	39	Lesen aus einem FIFO-Speicher	6.5.9

Tab. 6-16: Übersicht der Anweisungen FNC 30 bis 39

6.5.1 Rotation nach rechts (ROR)

		ROR		FNC 30			
		Rotation nach rechts					
		CPU	FX1S	FX1N	FX2N		
					●		
Operanden	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnY, KnM, KnS, T, C, D, V, Z ^①	K, H ^②	●	16 Bit	32 Bit	ROR, RORP	5
				●	●	DROR, DRORP	9

- ① Kn = K4 (16-Bit-Operation), Kn = K8 (32-Bit-Operation)
- ② n ≤ 16 (16-Bit-Operation), n ≤ 32 (32-Bit-Operation)

Funktionsweise

Rotieren von Bits um (n) Stellen nach rechts

Beschreibung

- Das Bit-Muster in (D+) wird mit jeder Ausführung von ROR um n Stellen nach rechts rotiert.
- Der Status des zuletzt rotierten Bits wird in M8022 (Carry Flag) kopiert.

HINWEIS

Wenn keine Flankenerkennung programmiert wird, wird das Bit-Muster in jedem Zyklus rotiert.

Beispiel ▾

ROR-Anweisung

Die Bitdaten im Datenregister D0 werden jedesmal um 4 Bits (K4) nach rechts verschoben, wenn der Eingang X0 von AUS nach EIN schaltet. Der Wert des zuletzt rotierenden Bits wird im Carry Flag gespeichert.

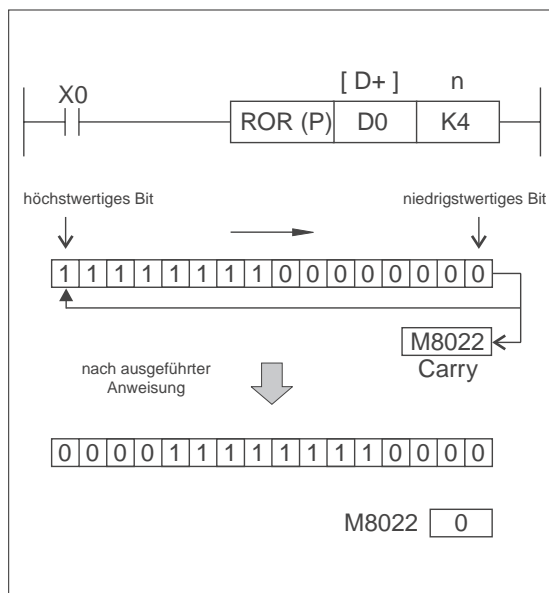


Abb. 6-56:
 Programmierbeispiel für eine Rotation nach rechts

C00091C



6.5.2 Rotation nach links (ROL)

			ROL		FNC 31			
			Rotation nach links					
Operanden			Puls-Anweisung (P)	Verarbeitung		Programmschritte		
				CPU	FX1S	FX1N	FX2N	
D+ KnY, KnM, KnS, T, C, D, V, Z ^①			n K, H ^②	●	16 Bit	32 Bit	ROL, ROLP	5
				●	●	●	DROL, DROLP	9

- ① Kn = K4 (16-Bit-Operation), Kn = K8 (32-Bit-Operation)
- ② n ≤ 16 (16-Bit-Operation), n ≤ 32 (32-Bit-Operation)

Funktionsweise

Rotieren von Bits um n Stellen nach links

Beschreibung

- Das Bit-Muster in (D+) wird mit jeder Ausführung von ROR um n Stellen nach links rotiert.
- Der Status des zuletzt rotierten Bits wird in M8022 (Carry Flag) kopiert.

HINWEIS

Wenn keine Flankenerkennung programmiert wird, wird das Bit-Muster in jedem Zyklus rotiert.

Beispiel ▾

ROL-Anweisung

Die Bitdaten im Datenregister D0 werden jedesmal um 4 Bits (K4) nach links verschoben, wenn der Eingang X0 von AUS nach EIN schaltet. Der Wert des zuletzt rotierenden Bits wird im Carry Flag gespeichert.

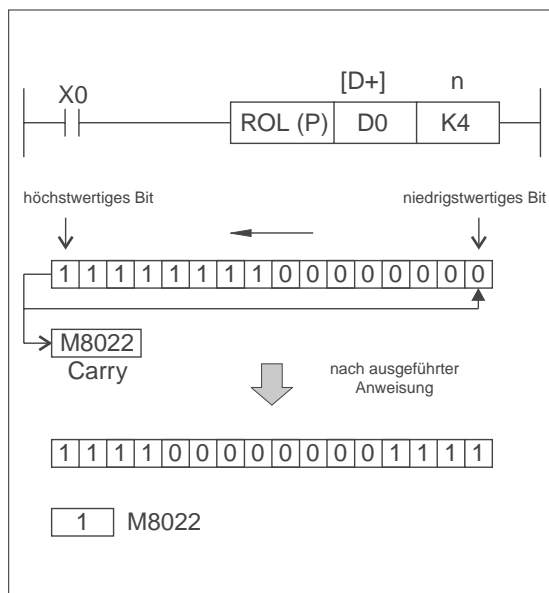


Abb. 6-57:
Programmierbeispiel für eine Rotation nach links

C000092C



6.5.3 Rotieren von Bits nach rechts (RCR)

		RCR		FNC 32			
		Rotieren von Bits nach rechts					
Operanden	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
				16 Bit	32 Bit		RCR, RCRP
	KnY, KnM, KnS, T, C, D, V, Z ^①	K, H ^②	●	●	●	DRCR, DRCRP	9

- ① Kn = K4 (16-Bit-Operation), Kn = K8 (32-Bit-Operation)
- ② n ≤ 16 (16-Bit-Operation), n ≤ 32 (32-Bit-Operation)

Funktionsweise:

Rotieren von Bits um n Stellen nach rechts unter Einbeziehung des Carry-Flags M8022

Beschreibung:

- Das Bit-Muster in (D+) wird um n Stellen nach rechts rotiert.
- Das Carry-Flag M8022 wird in die Rotationsschleife eingefügt.
- Wenn in (D+) ein aus Bits zusammengefaßter Operand genutzt werden soll, sind für die Zusammenfassung nur die Konstanten K4 (16-Bit-Operation) und K8 (32-Bit-Operation) gültig.

- HINWEIS** | Wenn keine Flankenerkennung programmiert wird, wird die Rotation in jedem Programmzyklus ausgeführt.
- | Der Status des Carry-Bits beim Einschalten der Anweisung wird mit in das zu rotierende Bit-Muster aufgenommen.

Beispiel ▾ RCR-Anweisung

Die Bit-Daten im Datenregister D0 werden jedesmal um 4 Bits (K4) nach rechts verschoben, wenn der Eingang X0 von AUS nach EIN schaltet.

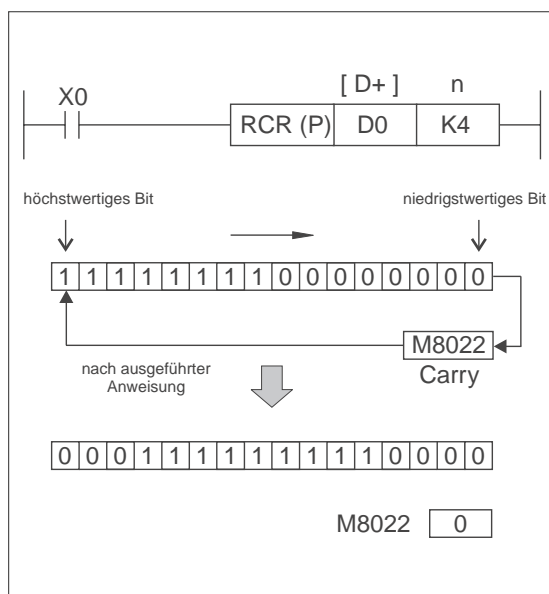


Abb. 6-58:
Programmierbeispiel für eine Rotation nach rechts

C000093C

6.5.4 Rotieren von Bits nach links (RCL)

		RCL		FNC 33					
		Rotieren von Bits nach links							
Operanden D+ KnY, KnM, KnS, T, C, D, V, Z ^①		n K, H ^②		Puls-Anweisung (P)		Verarbeitung		Programmschritte	
				●		16 Bit	32 Bit	RCL, RCLP	5
						●	●	DRCL, DRCLP	9

- ① Kn = K4 (16-Bit-Operation), Kn = K8 (32-Bit-Operation)
- ② n ≤ 16 (16-Bit-Operation), n ≤ 32 (32-Bit-Operation)

Funktionsweise

Rotieren von Bits um n Stellen nach links unter Einbeziehung des Carry-Flags M8022

Beschreibung

- Das Bit-Muster in (D+) wird um n Stellen nach links rotiert.
- Das Carry-Flag M8022 wird in die Rotationsschleife eingefügt.
- Wenn in (D+) ein aus Bits zusammengefaßter Operand genutzt werden soll, sind für die Zusammenfassung nur die Konstanten K4 (16-Bit-Operation) und K8 (32-Bit-Operation) gültig.

HINWEIS | Wenn keine Flankenerkennung programmiert wird, wird die Rotation in jedem Programmzyklus ausgeführt.

| Der Status des Carry-Bits beim Einschalten der Anweisung wird mit in das zu rotierende Bit-Muster aufgenommen.

Beispiel ▾ RCL-Anweisung

Die Bit-Daten im Datenregister D0 werden jedesmal um 4 Bits (K4) nach links verschoben, wenn der Eingang X0 von AUS nach EIN schaltet.

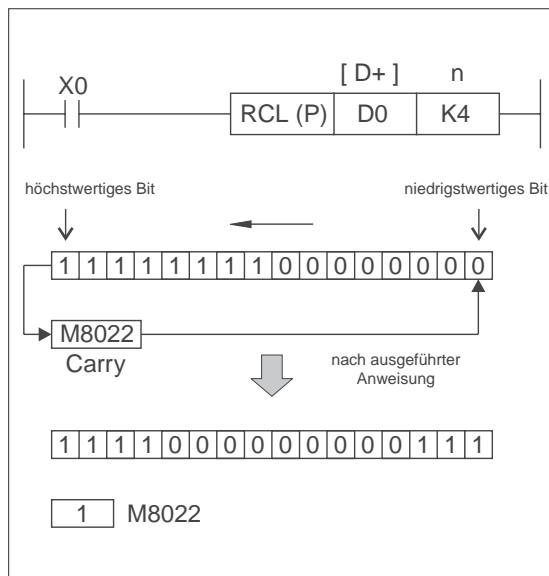


Abb. 6-59: Programmierbeispiel für eine Rotation nach links

C000094C

6.5.5 Binäre Daten bitweise verschieben (SFTR, SFTL)

				SFTR		FNC 34		
				Binäre Bitdaten bitweise verschieben, rechts				CPU
				FX1S	FX1N	FX2N		
				●	●	●		
Operanden	S+	D+	n1, n2	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S	Y, M, S	K, H		●	16 Bit	32 Bit	SFTR/ SFTRP
				●	●			

				SFTL		FNC 35		
				Binäre Bitdaten bitweise verschieben, links				CPU
				FX1S	FX1N	FX2N		
				●	●	●		
Operanden	S+	D+	n1, n2	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S	Y, M, S	K, H		●	16 Bit	32 Bit	SFTL/ SFTLP
				●	●			

Funktionsweise:

Binäre Daten werden bitweise nach rechts oder links verschoben.

Beschreibung:

- Diese Anweisung bezieht sich auf Bitoperanden, die zu einem Datenwort zusammengefaßt sind. Die Wortbreite wird mit n1 festgelegt.
- Mit der Ausführung der Anweisung werden die Bits in (D+) um n2 Stellen verschoben, und ausgehend von (S+) werden n2 Bits in (D+) in Abhängigkeit der Verschieberichtung eingefügt.
- n1: Anzahl der Zieladressen, beginnend mit der Startadresse in (D+)
n2: Anzahl der zu verschiebenden Bits
(n2 ≤ n1 ≤ 512) und (n1 ≤ maximal mögliche Adresse des in (D+) angegebenen Operanden)
- Mit der SFTR-Anweisung können Sie Daten bitweise nach rechts verschieben.
- Mit der SFTL-Anweisung können Sie Daten bitweise nach links verschieben.

HINWEIS

Die Anweisungen werden in jedem Programmzyklus ausgeführt. Dies können Sie durch Einsatz einer vorgeschalteten Impulsfunktion (PLS- oder PLF-Anweisung) oder die Verwendung des Befehlsparameters P verhindern.

Beispiel ▾ Einsatz der SFTR-Anweisung

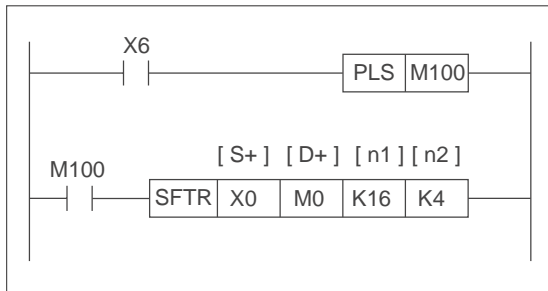


Abb. 6-60: Programmierbeispiel zum Einsatz der SFTR-Anweisung

C000090C

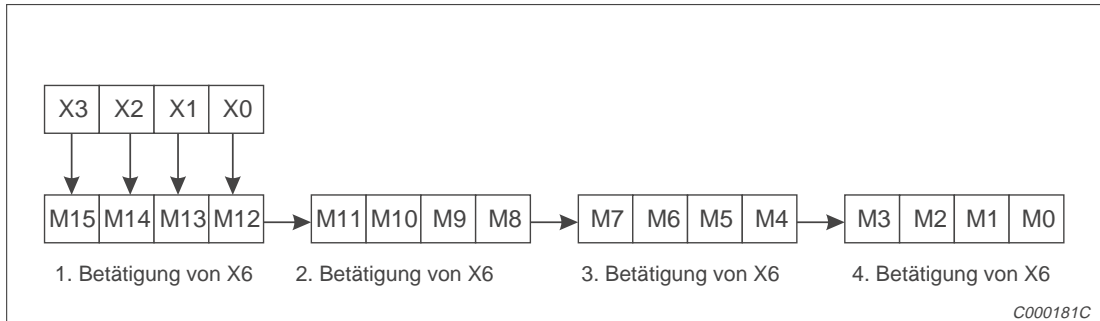


Abb. 6-61: Beispiel zum bitweisen Verschieben nach rechts

Mit der Ausführung von X6 werden die an den Eingängen X0 bis X3 anliegenden binären Signale bitweise in den definierten Merkerbereich eingelesen und entsprechend nach rechts verschoben. △

Beispiel ▾ Einsatz der SFTL-Anweisung

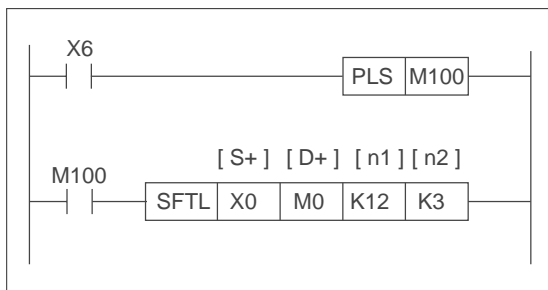


Abb. 6-62: Programmierbeispiel zum Einsatz der SFTL-Anweisung

C000116C

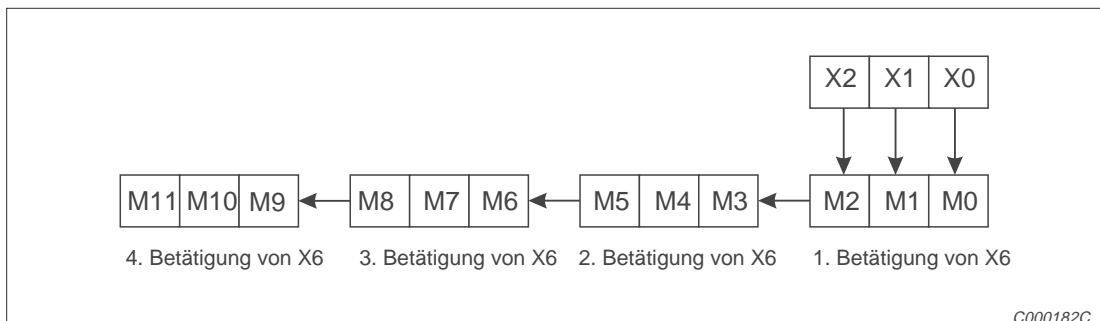


Abb. 6-63: Beispiel zum bitweisen Verschieben nach links

Wird X6 betätigt, werden die binären Signale an den Eingängen X0 bis X2 bitweise in den definierten Merkerbereich eingelesen und nach links verschoben. △

6.5.6 Daten wortweise nach rechts verschieben (WSFR)

				WSFR		FNC 36	
				Daten wortweise nach rechts verschieben			
				CPU	FX1S	FX1N	FX2N
							●
Operanden	S+	D+	n1, n2	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	KnX,KnY,KnM, KnS,T,C,D	KnY,KnM,KnS, T,C,D	K, H n2≤n1≤512	●	16 Bit ●	32 Bit	WSFR/ WSFRP
							9

Funktionsweise

Daten werden wortweise nach rechts verschoben.

Beschreibung

- Die Quelldaten (S+) werden in einen Stapelspeicher (D+) geschrieben und verschoben. Die Tiefe des Stapels beträgt n1 Worte.
- Mit jeder Ausführung der Anweisung werden n2 Worte eingelesen und der Inhalt des Stapels verschoben.

HINWEIS | Bei der Verwendung von zusammengefaßten Bit-Operanden ist darauf zu achten, daß (S+) und (D+) über dieselbe Anzahl von Bits verfügen.

Beispiel ▾ WSFR-Anweisung

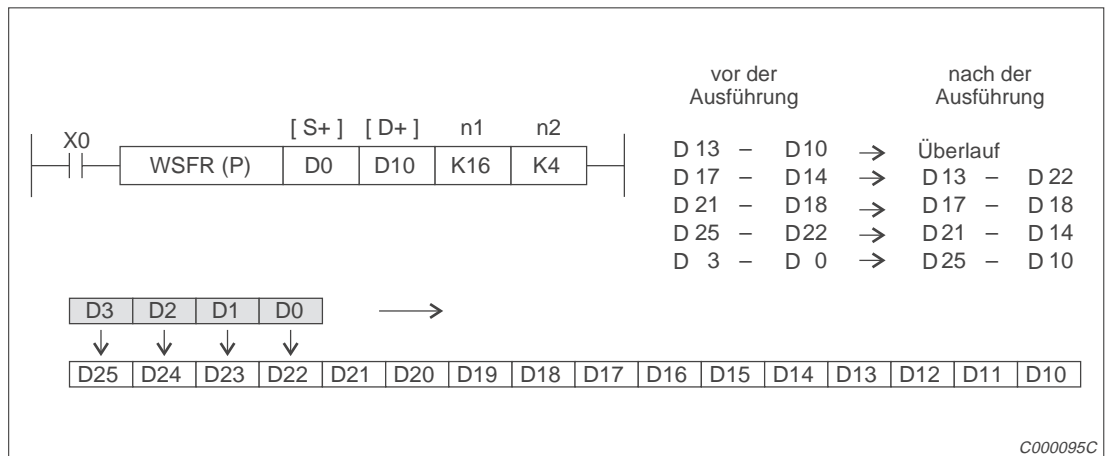


Abb. 6-64: Programmierbeispiel zum Verschieben nach rechts



6.5.7 Daten wortweise nach links verschieben (WSFL)

				WSFL		FNC 37					
				Daten wortweise nach links verschieben							
Operanden				CPU	FX1S	FX1N	FX2N				
							●				
S+		D+		n1, n2		Puls-Anweisung (P)		Verarbeitung		Programmschritte	
KnX,KnY,KnM,KnS,T,C,D		KnY,KnM,KnS,T,C,D		K, H n2≤n1≤512		●		16 Bit	32 Bit	WSFL/WSFLP	
								●		9	

Funktionsweise

Daten werden wortweise nach links verschoben.

Beschreibung

- Die Quelldaten (S+) werden in einen Stapelspeicher (D+) geschrieben und verschoben. Die Tiefe des Stapels beträgt n1 Worte.
- Mit jeder Ausführung der Anweisung werden n2 Worte eingelesen und der Inhalt des Stapels verschoben.

HINWEIS | Bei der Verwendung von zusammengefaßten Bit-Operanden ist darauf zu achten, daß (S+) und (D+) über dieselbe Anzahl von Bits verfügen.

Beispiel ▾ WSFL-Anweisung

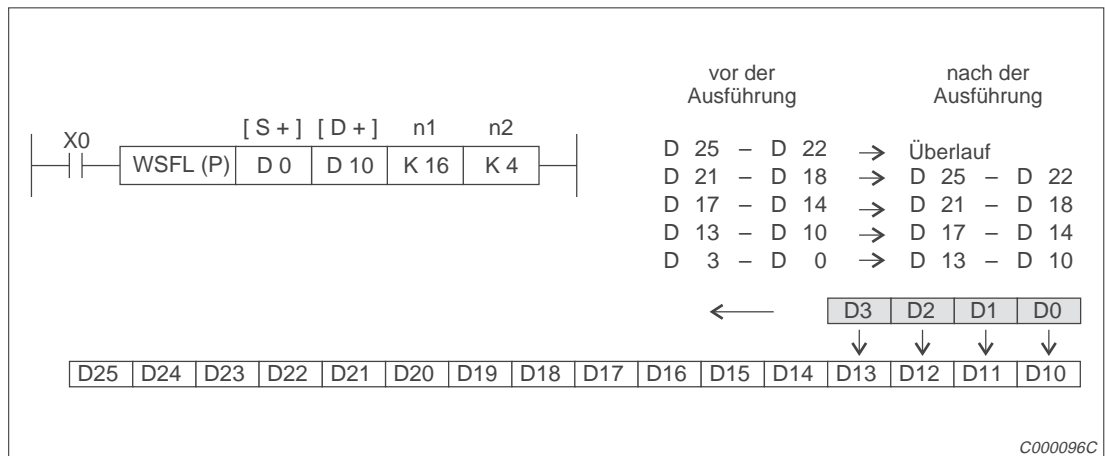


Abb. 6-65: Programmierbeispiel zum Verschieben nach links

6.5.8 Schreiben in einen FIFO-Speicher (SFWR)

				SFWR		FNC 38		
				Schreiben in einen FIFO-Speicher				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX,KnY,KnM,KnS,T,C,D,V,Z	KnY,KnM,KnS,T,C,D	K, H 2≤n≤512	●	16 Bit	32 Bit	SFWR/SFWRP	7

Funktionsweise

Einlesen von Wörtern in einen durch die Anweisung definierten FIFO-Speicher.

Beschreibung

- Die Daten aus (S+) werden in einen Stapelspeicher geschrieben.
- Die erste Adresse des Stapels ist (D+).
- Die Tiefe des Stapels beträgt (n) Worte.
- Im Stapel können maximal (n-1) Worte abgelegt werden, da (D+) als Zeiger für den Stapel verwendet wird. (D+) sollte vor der ersten Ausführung auf Null gesetzt werden.
- Wenn (n-1) Worte in den Stapel eingetragen wurden, ohne andere Worte auszulesen, ist es nicht möglich, weitere Worte einzutragen. Dieser Zustand wird durch das Einschalten des Carry-Bits (M8022) angezeigt.
- Mit jeder Ausführung der Anweisung wird der Zeiger (D+) inkrementiert.
- Die Anweisung wird zusammen mit der Anweisung SFRD eingesetzt; der Parameter (n) sollte in beiden Anweisungen gleich sein.

Beispiel ▾

SFWR-Anweisung

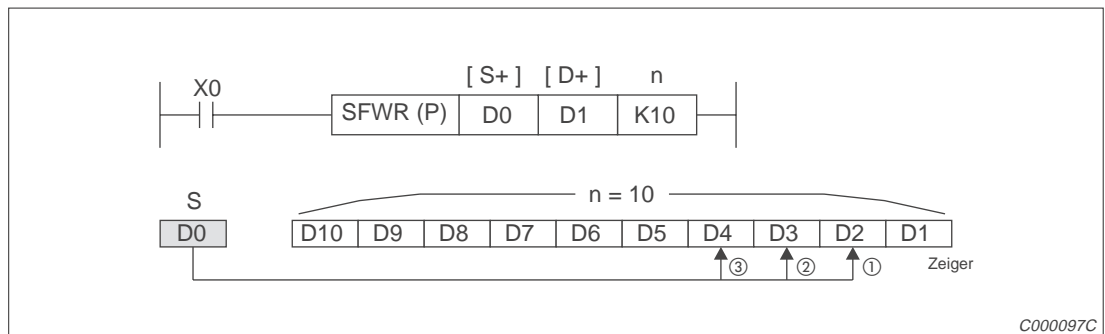


Abb. 6-66: Programmierbeispiel zum Schreiben in einen FIFO-Speicher

Anwendungsbeispiel siehe Seite 6-74.



6.5.9 Lesen aus einem FIFO-Speicher (SFRD)

				SFRD		FNC 39																			
				Lesen aus einem FIFO-Speicher																					
<table border="1"> <tr> <th>Operanden</th> <th>S+</th> <th>D+</th> <th>n</th> <th>Puls-Anweisung (P)</th> <th colspan="2">Verarbeitung</th> <th colspan="2">Programmschritte</th> </tr> <tr> <td></td> <td>KnX,KnY,KnM,KnS,T,C,D,V,Z</td> <td>KnY,KnM,KnS,T,C,D</td> <td>K, H 2 ≤ n ≤ 512</td> <td>●</td> <td>16 Bit</td> <td>32 Bit</td> <td>SFRD/SFRDP</td> <td>7</td> </tr> </table>				Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte			KnX,KnY,KnM,KnS,T,C,D,V,Z	KnY,KnM,KnS,T,C,D	K, H 2 ≤ n ≤ 512	●	16 Bit	32 Bit	SFRD/SFRDP	7	CPU	FX1S	FX1N	FX2N
				Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte														
	KnX,KnY,KnM,KnS,T,C,D,V,Z	KnY,KnM,KnS,T,C,D	K, H 2 ≤ n ≤ 512	●	16 Bit	32 Bit	SFRD/SFRDP	7																	
					●		●																		

Funktionsweise

Auslesen von Wörtern aus einem FIFO-Speicher

Beschreibung

- Aus dem mit (S+) beginnenden Stapelspeicher wird der Inhalt von ((S+)+1) nach (D+) ausgelesen.
- Der Zeiger des Stapelspeichers (S+) wird mit jeder Ausführung von SFRD dekrementiert.
- Die Werte in ((S+)+2) bis ((S+)+n) werden um eine Position aufwärts geschoben.
- Wenn (S+) den Wert Null annimmt, ist der Stapelspeicher leer. Dies wird durch das Bit M8020 angezeigt.
- Die Anweisung SFRD arbeitet zusammen mit der Anweisung SFWR. Der Parameter n sollte in beiden Anweisungen gleich sein.

Beispiel ▾

SFRD-Anweisung

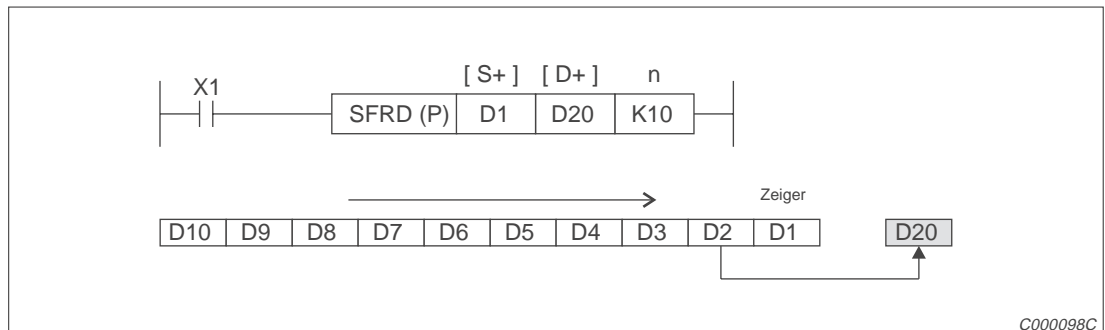


Abb. 6-67: Programmierbeispiel zum Lesen aus einem FIFO-Speicher

Anwendungsbeispiel siehe Seite 6-74.



Beispiel ▾ Programmierung eines FIFO-Speichers

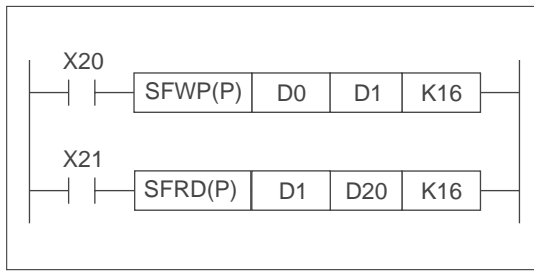


Abb. 6-68:
Programmierbeispiel zum Schreiben und Lesen eines FIFO-Speichers

C000146C

Es wird ein Stapelspeicher mit 15 Adressen und einem Pointer definiert.

Betätigen von:	/	X20	X20	X20	X21	X20	X21
D20	0	0	0	0	55	55	66
D0	0	55	66	77	77	88	88
D1	0	1	2	3	2	3	2
D2	0	55	55	55	66	66	77
D3	0	0	66	66	77	77	88
D4	0	0	0	77	0	88	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
D15	0	0	0	0	0	0	0
D16	0	0	0	0	0	0	0

Abb. 6-69: Register des FIFO-Speichers

Mit jeder Betätigung von X20 wird der Zahlenwert von D0 an die erste noch unbelegte Adresse innerhalb des Stapelspeichers geschrieben.

Mit jeder Betätigung von X21 wird der Inhalt von D2 nach D20 ausgelesen und der Inhalt der anderen Adressen innerhalb des Stapelspeichers um eine Position vorgeschoben. △

6.6 Datenoperationen

Übersicht der Anweisungen FNC 40 bis 49

Symbol	FNC	Bedeutung	Abschnitt
ZRST	40	Operandenbereiche zurücksetzen	6.6.1
DECO	41	Daten decodieren	6.6.2
ENCO	42	Daten codieren	6.6.3
SUM	43	Ermittlung gesetzter Bits	6.6.4
BON	44	Überprüfung eines Bits	6.6.5
MEAN	45	Ermittlung eines Durchschnittswertes	6.6.6
ANS	46	Starten eines Zeitintervalls	6.6.7
ANR	47	Rücksetzen von Anzeigebits	6.6.8
SQR	48	Ermittlung der Quadratwurzel	6.6.9
FLT	49	Umwandlung des Zahlenformats	6.6.10

Tab. 6-17: Übersicht der Anweisungen FNC 40 bis 49

6.6.1 Operandenbereiche zurücksetzen (ZRST)

		ZRST		FNC 40	
		Operandenbereiche zurücksetzen			
Operanden		CPU	FX1S	FX1N	FX2N
			●	●	●
Operanden	D1+, D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	Y, M, S, T, C, D	●	16 Bit	32 Bit	ZRST/ ZRSTP

Funktionsweise

Mehrere aufeinanderfolgende Operanden (Operandenbereiche) lassen sich mit nur einer ZRST-Anweisung auf den Signalzustand oder Istwert „0“ zurücksetzen.

Beschreibung

- In (D1+) und (D2+) wird der Operandenbereich festgelegt, den Sie zurücksetzen möchten.
- In (D1+) und (D2+) müssen Sie den gleichen Operandentyp angeben.

(D1+): Erste Operandenadresse
 (D2+): Letzte Operandenadresse

Es muß gelten: (D1+) ≤ (D2+)

Wenn (D1+) > (D2+), wird nur der in (D1+) angegebene Operand zurückgesetzt.

HINWEISE

Obwohl es sich um eine 16-Bit-Operation handelt, können in den beiden Zieladressen auch 32-Bit-Counter eingesetzt werden. Der kombinierte Einsatz von 16- und 32-Bit-Countern ist jedoch nicht zulässig. So ist es z.B. nicht erlaubt, in (D1+) einen 16-Bit-Counter und in (D2+) einen 32-Bit-Counter anzugeben.

Einzelne Operanden können mit der RST-Anweisung zurückgesetzt werden (siehe auch Abs. 4.10).

Beispiel ▾

Einsatz der ZRST-Anweisung

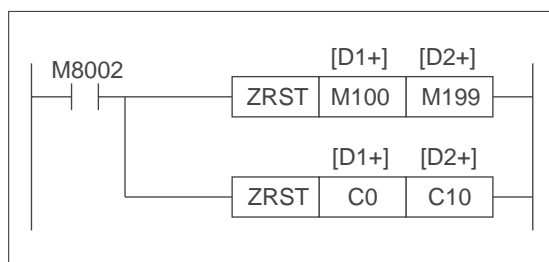


Abb. 6-70:

Programmierbeispiel zum Einsatz der ZRST-Anweisung

C000100C

Die Bit-Operanden M100 bis M199 werden auf den Signalzustand „0“ zurückgesetzt. Die Wortoperanden C0 bis C10 werden auf den Istwert „0“ zurückgesetzt. Die zugehörigen Spulen und Kontakte werden ausgeschaltet. △

6.6.2 Daten decodieren (DECO)

				DECO		FNC 41		
				Daten decodieren				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, X, Y, M, S, T, C, D, V, Z	Y, M, S, T, C, D	K, H		●	16 Bit	32 Bit	DECO/ DECOP

Funktionsweise

Daten decodieren

Beschreibung

Die Daten von n Operanden, ausgehend von der in (S+) angegebenen Startadresse, werden decodiert. In (D+) wird die Startadresse der Zieloperanden festgelegt, in denen das Decodierergebnis abgelegt wird.

n: Anzahl der Operanden, deren Daten decodiert werden sollen.
 Bei der Angabe eines Bit-Operanden in D+ muß gelten: (1 ≤ n ≤ 8).
 Bei der Angabe eines Wortoperanden in D+ muß gelten: (1 ≤ n ≤ 4).

(S+): Startadresse der Operanden, deren Daten decodiert werden sollen.

2ⁿ: Anzahl der Zieloperanden

(D+): Startadresse der Zieloperanden

HINWEISE

Die Anweisung wird nicht ausgeführt, wenn n = 0 ist.

Die Anweisung wird nur ausgeführt, wenn die Eingangsbedingung gesetzt ist. Der zugehörige Ausgang bleibt aktiviert, auch wenn die Eingangsbedingung anschließend wieder ausgeschaltet wird.

Fehlerquellen

- Ein Programmablauffehler tritt auf, wenn n nicht im Bereich von 0 bis 8 liegt.
- Ein Programmablauffehler tritt auf, wenn sämtliche Bits der Ausgangsoperanden den Wert 0 aufweisen.

Beispiel ▾

Einsatz der DECO-Anweisung mit Angabe eines Bit-Operanden in D+ ($1 \leq n \leq 8$)

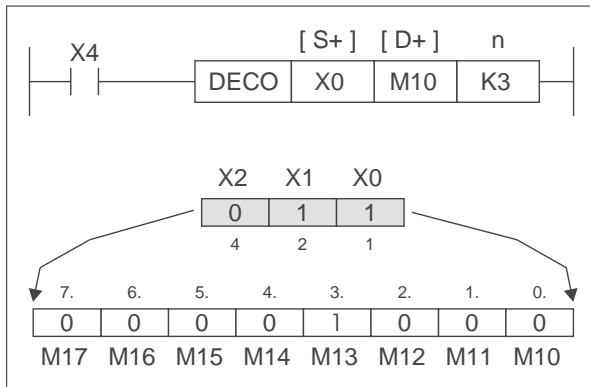


Abb. 6-71:
 Programmierbeispiel zum Einsatz der DECO-Anweisung mit Angabe eines Bit-Operanden in D+

C000101C

Wenn $n = 3$, lauten die Eingangsoperanden X_0 , X_1 und X_2 . Weil $2^n = 2^3 = 8$, stehen als Zieladressen die Merker M_{10} bis M_{17} zur Verfügung.

Der Wert der Eingangsoperanden ist $1+2=3$. Entsprechend wird das 3. Bit der Zieladressen, d.h. der Merker M_{13} , eingeschaltet. Lautet der Wert der Eingangsoperanden 0, wird der Merker M_{10} eingeschaltet. △

6.6.3 Daten codieren (ENCO)

				ENCO		FNC 42	
				Daten codieren			
				CPU	FX1S	FX1N	FX2N
				●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	X, Y, M, S, T, C, D, V, Z	T, C, D, V, Z	K, H	●	16 Bit ●	32 Bit	ENCO/ ENCOP
							7

Funktionsweise

Daten codieren

Beschreibung

Die Daten von 2^n Operanden, ausgehend von der in (S+) angegebenen Startadresse, werden codiert. In (D+) wird der Zieloperand festgelegt, in dem das Codierergebnis abgelegt wird.

2^n : Anzahl der Operanden, deren Daten codiert werden sollen.

n: Anzahl der Zieloperanden

Bei der Angabe eines Bit-Operanden in (S+) muß gelten: $(1 \leq n \leq 8)$

Bei der Angabe eines Wortoperanden in (S+) muß gelten: $(1 \leq n \leq 4)$

(S+): Startadresse der Operanden, deren Daten codiert werden sollen.

(D+): Zieloperand

HINWEISE

Haben mehrere der in (S+) angegebenen Operanden den Wert 1, wird nur das höchste Bit bearbeitet.

Die Anweisung wird nicht ausgeführt, wenn $n = 0$ ist.

Die Anweisung wird nur ausgeführt, wenn die Eingangsbedingung gesetzt ist. Der zugehörige Ausgang bleibt aktiviert, auch wenn die Eingangsbedingung anschließend wieder ausgeschaltet wird.

Fehlerquellen

- Ein Programmablauffehler tritt auf, wenn n nicht im Bereich von 0 bis 8 liegt.
- Ein Programmablauffehler tritt auf, wenn sämtliche Bits der Ausgangsoperanden den Wert 0 aufweisen.

Beispiel ▾

Einsatz der ENCO-Anweisung mit Angabe eines Bit-Operanden in (S+) ($1 \leq n \leq 8$)

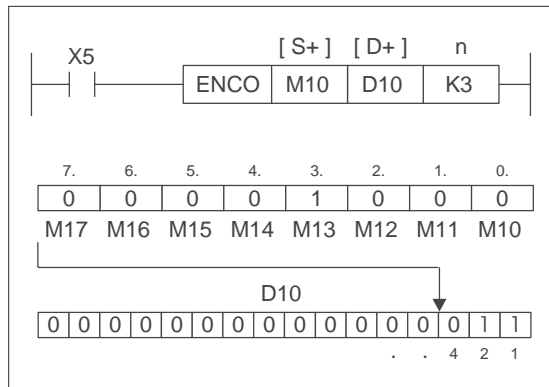


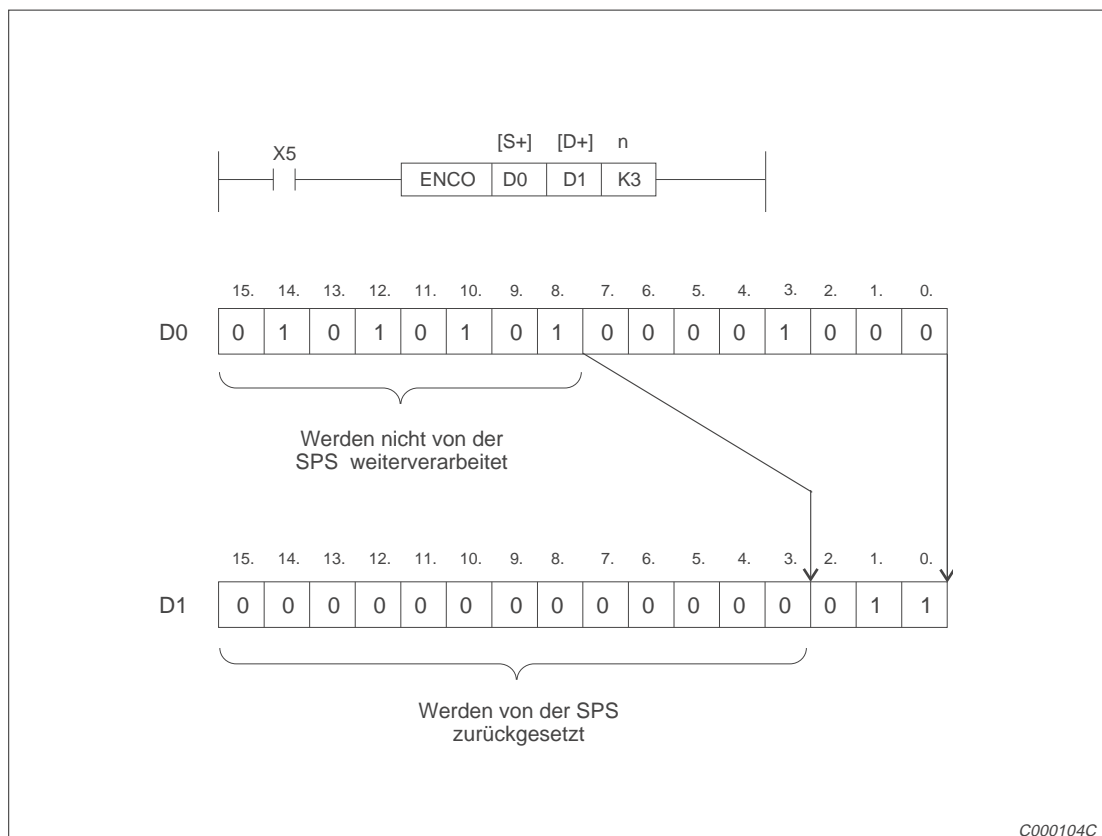
Abb. 6-73: Programmierbeispiel zum Einsatz der ENCO-Anweisung mit Angabe eines Bit-Operanden in (S+)

C000103C

Wenn $2^n = 2^3 = 8$, stehen als Ausgangsadressen die Merker M10 bis M17 zur Verfügung. Weil bei den Ausgangsoperanden der 3. Operand, d.h. der Merker M13 gesetzt ist, wird in das Datenregister D10 der Wert 3 geschrieben. ▴

Beispiel ▾

Einsatz der ENCO-Anweisung mit Angabe eines Wortoperanden in (S+) ($1 \leq n \leq 4$)




C000104C

Abb. 6-74: Programmierbeispiel zum Einsatz der ENCO-Anweisung mit Angabe eines Wortoperanden in (S+)

Im Datenregister D0 ist das 3. Bit gesetzt. Es wird also der Zahlenwert 3 codiert und im Datenregister D1 abgespeichert. ▴

6.6.4 Ermittlung gesetzter Bits (SUM)

		SUM		FNC 43			
		Ermittlung gesetzter Bits					
		CPU	FX1S	FX1N	FX2N		
					●		
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	SUM, SUMP	7
				●	●	DSUM, DSUMP	9

Funktionsweise

Ermitteln der Anzahl gesetzter Bits in einem Datenwort.

Beschreibung

- Es wird die Anzahl der eingeschalteten Bits in (S+) ermittelt.
- Der ermittelte Wert wird nach (D+) geschrieben.

HINWEIS

Wenn eine 32-Bit-Operation durchgeführt wird, werden die oberen 16 Bit ((D+)+1) des Zieloperanden (D+) auf Null gesetzt, da die maximale Anzahl eingeschalteter Bits in (S+) 32 beträgt.

Beispiel ▾

SUM-Anweisung

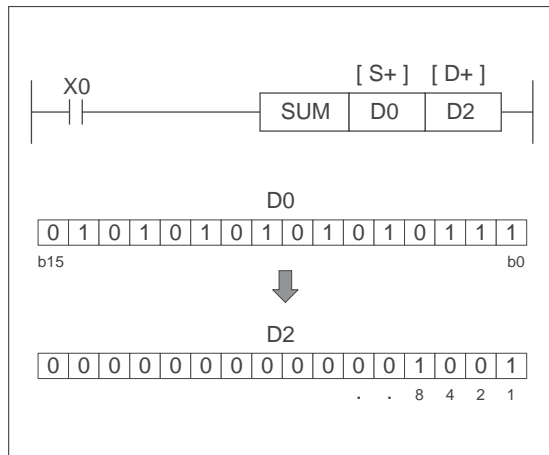


Abb. 6-75:

Programmierbeispiel zur Ermittlung gesetzter Bits

C000141C



6.6.5 Überprüfen eines Bits (BON)

				BON		FNC 44		
				Überprüfen eines Bits				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, S, T, C, D, V, Z	Y, M, S	K, H ^①	●	16 Bit	32 Bit	BON/BONP	7
					●	●	DBON/DBONP	13

① n = 0 bis 15 (16-Bit-Operation); n = 0 bis 31 (32-Bit-Operation)

Funktionsweise

Überprüfen eines einzelnen Bits innerhalb eines Datenwortes.

Beschreibung

- Wenn das Bit Nummer n innerhalb von (S+) eingeschaltet ist, wird (D+) eingeschaltet.

Beispiel ▾

BON-Anweisung

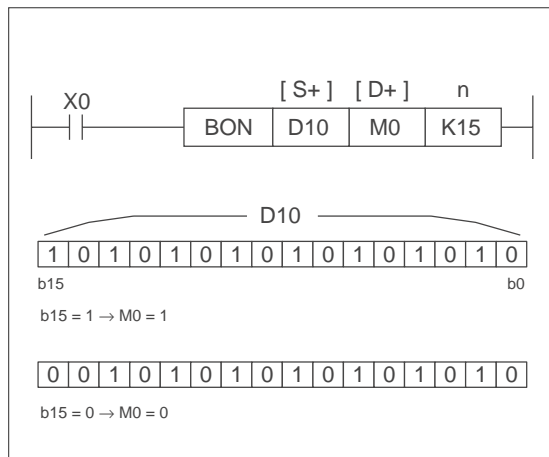


Abb. 6-76:
 Programmierbeispiel zum Überprüfen eines Bits

C000142C

6.6.6 Ermittlung von Durchschnittswerten (MEAN)

				MEAN		FNC 45		
				Ermittlung von Durchschnittswerten				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS, T, C, D	KnY, KnM, KnS, T, C, D, V, Z	K, H (n = 1 bis 64)		●	16 Bit	32 Bit	MEAN/ MEANP
					●	●	DMEAN/ DMEANP	13

Funktionsweise

Bilden des arithmetischen Mittelwerts aus mehreren Datenwörtern.

Beschreibung

Ausgehend von (S+) werden n Datenwörter addiert und durch n dividiert. Das ganzzahlige Ergebnis wird nach (D+) geschrieben.

HINWEIS | Wird n größer als der verfügbare Operandenbereich, ausgehend von (S+), gewählt, so wird n automatisch auf die verfügbare Anzahl von Operanden angepasst.

Fehlerquelle

Wenn (n) außerhalb des Bereichs (1-64) liegt, wird ein Fehler erzeugt.

Beispiel ▾ MEAN-Anweisung

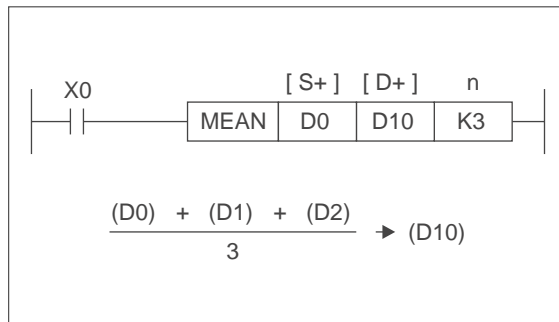
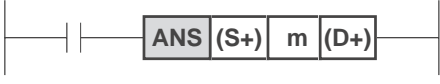


Abb. 6-77:
 Programmierbeispiel zur Ermittlung von Durchschnittswerten

C000143C



6.6.7 Starten eines Zeitintervalls (ANS)

				ANS		FNC 46		
				Starten eines Zeitintervalls				CPU
Operanden	S+	D+	m	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T T0 bis T199	S S900 bis S999	K (1 bis 32767)		16 Bit	32 Bit	ANS	7

Funktionsweise

Starten eines Zeitintervalls und Schalten eines Anzeige-Bits.

Beschreibung

- Die Startoperanden S900 bis S999 können als Anzeige-Bits verwendet werden.
- Mit der Ausführung dieser Anweisung wird eine Zeit von m x 100 ms gestartet.
- Nach Ablauf der Zeit wird das Anzeige-Bit (D+) eingeschaltet.
- In (S+) wird ein Timer vorgegeben, der das Zeitintervall bildet.

HINWEIS | Der verwendete Timer darf im weiteren Programm nicht mehr eingesetzt werden.

Beispiel ▾ Programmierung der ANS-Anweisung

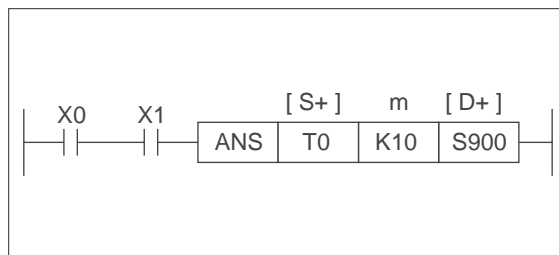
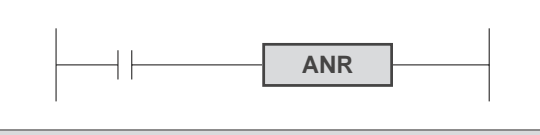


Abb. 6-78:
 Programmierbeispiel zum Starten eines Zeitintervalls

C000144C



6.6.8 Rücksetzen von Anzeige-Bits (ANR)

		ANR		FNC 47		
		Rücksetzen eines Anzeige-Bits				
		CPU	FX1S	FX1N	FX2N	
					●	
Operanden	—	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	—		●	16 Bit	32 Bit	ANR/ANRP

Funktionsweise

Rücksetzen von Anzeige-Bits

Beschreibung

Wenn die Anweisung aktiv ist, wird das aktive Anzeige-Bit mit der niedrigsten Adresse zurückgesetzt.

HINWEIS | Die Anweisung sollte mit der Option „P“ ausgeführt werden.

Beispiel ▾ Programmierung der ANR-Anweisung

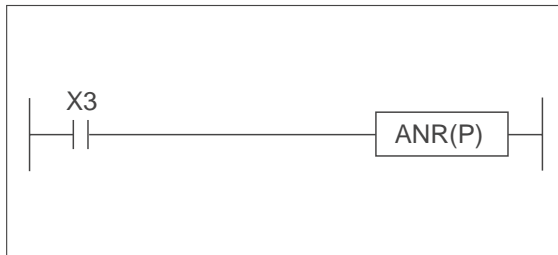


Abb. 6-79:
 Programmierbeispiel zum Rücksetzen von Anzeige-Bits

C000145C

Wird X3 eingeschaltet, wird das gesetzte Anzeigebit zwischen S900 und S999 zurückgesetzt. Wurden mehrere Anzeigebits gesetzt, wird das Anzeigebit mit der niedrigeren Adresse zurückgesetzt.

Die anderen gesetzten Anzeigebits werden durch nochmaliges Einschalten von X3 in aufsteigender Reihenfolge zurückgesetzt. △

6.6.9 Ermittlung der Quadratwurzel (SQR)

			SQR		FNC 48	
			Berechnung der Quadratwurzel			
Operanden			CPU	FX1S	FX1N	FX2N
						●
S+			D+			Puls-Anweisung (P)
K, H, D			D			●
			Verarbeitung		Programmschritte	
			16 Bit	32 Bit	SQR/SQRP	5
			●	●	DSQR/DSQRP	9

Funktionsweise

Berechnung der Quadratwurzel, (D+) =

Beschreibung

Ausgehend von (S+) wird die Quadratwurzel berechnet und auf einen ganzzahligen Wert abgerundet nach (D+) geschrieben.

Beispiel ▾ Programmierung der SQR-Anweisung ohne Flag

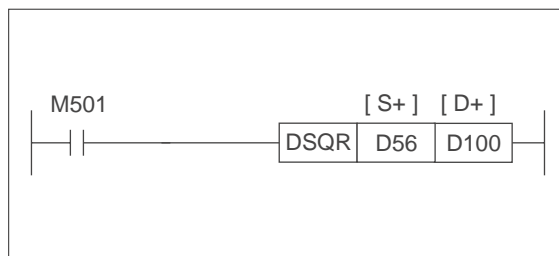


Abb. 6-88:
Programmierbeispiel zur Berechnung der Quadratwurzel

C000302C

Wird der Merker M501 eingeschaltet, wird vom Wert im Datenregister D56 die Quadratwurzel berechnet, und das Ergebnis als abgerundeter ganzzahliger Wert nach Datenregister D100 geschrieben.

HINWEIS Die Wurzel aus einer negativen Zahl führt immer zu einem Fehler und Fehlermerker M8067 wird eingeschaltet.

In der folgenden Tabelle sind einige Beispielergebnisse für die Wurzelberechnung SQR aufgeführt.

(S+)	Ergebnis	(D+)
25	5,0	5
60	7,746	7
-236	15,36 i	ERROR
147	12,124	12

Tab. 6-20:
Beispielergebnisse zur Wurzelberechnung

△

6.6.10 Umwandlung des Zahlenformats (FLT)

			FLT		FNC 49		
			Umwandlung des Zahlenformats				
			CPU	FX1S	FX1N	FX2N	
						●	
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D	D		●	16 Bit	32 Bit	FLT/FLTP
				●	●	DFLT/DFLTP	9

Funktionsweise

Umwandlung einer Zahl vom Integer-Format in das Fließkomma-Format

Beschreibung

Ausgehend von (S+) wird die Zahl in eine Fließkomma-Zahl umgewandelt und nach (D+) geschrieben.

HINWEIS | Das Ergebnis der Zahlenumwandlung wird immer in einem 32-Bit-Datenregister abgelegt.

Beispiel ▾ Programmierung der FLT-Anweisung

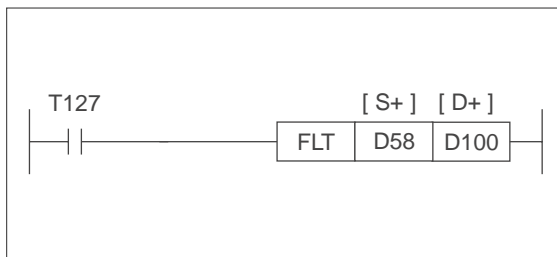


Abb. 6-89:
 Programmierbeispiel zur
 Umwandlung in Fließkomma-Format

C000304C

Sobald der Timer T127 einschaltet, wird der Wert im Datenregister D58 in eine Fließkomma-Zahl umgewandelt, und der Wert nach Datenregister D100 geschrieben.

△

6.7 High-Speed-Anweisungen

Übersicht der Anweisungen FNC 50 – 59

Symbol	FNC	Bedeutung	Abschnitt
REF	50	Ein- und Ausgänge auffrischen	6.7.1
REFF	51	Einstellen der Eingangsfiler	6.7.2
MTR	52	Einlesen einer Matrix	6.7.3
DHSCS	53	Setzen durch High-Speed-Counter	6.7.4
DHSCR	54	Rücksetzen durch High-Speed-Counter	6.7.4
DHSZ	55	Bereichsvergleich	6.7.5
SPD	56	Geschwindigkeitserkennung	6.7.6
PLSY	57	Ausgabe einer definierten Anzahl von Impulsen	6.7.7
PWM	58	Impulsausgabe mit Modulation der Impulsweite	6.7.8
PLSR	59	Ausgabe einer bestimmten Anzahl von Impulsen	6.7.9

Tab. 6-21: Übersicht der Anweisungen FNC 50–59

6.7.1 Ein- und Ausgänge auffrischen (REF)

		REF		FNC 50			
		Ein- und Ausgänge auffrischen					
		CPU	FX1S	FX1N	FX2N		
			●	●	●		
Operanden	D	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y ①	K, H ②	●	16 Bit	32 Bit	REF, REFP	5

① Die Operanden sollten ein Vielfaches von 10 sein: X0, X10, X20, etc.

② n muß ein Vielfaches von 8 sein: 8, 16, 24, etc.

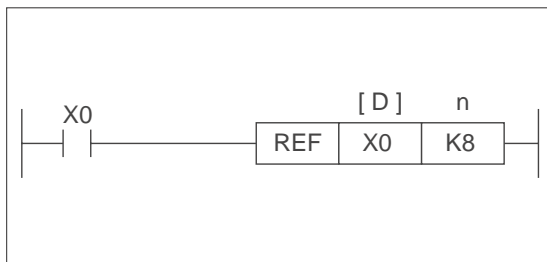
Funktionsweise

Ein- und Ausgänge auffrischen (aktualisieren)

Beschreibung

- Die Programmabarbeitung bei den Steuerungen der FX-Serie erfolgt nach dem Prozessabbildverfahren. Vor der Programmabarbeitung liest die Steuerung die Eingangssignalzustände und speichert sie im Eingangsimageregister. Es werden also nicht die Eingänge, sondern das Eingangsimageregister verarbeitet.
Nach der Programmabarbeitung werden die Daten des Ausgangsimageregisters gelesen und an den Ausgabespeicher übertragen.
- Mit der REF-Anweisung können die Eingänge während eines Programmzyklus abgefragt und die Imageregister aufgefrischt (aktualisiert) werden.
- Die REF-Anweisung können Sie verwenden, um die letzte Eingangsinformation zu lesen, während eine Operation ausgeführt wird.
- Desweiteren kann mit der REF-Anweisung das Operationsergebnis unmittelbar nach der Abarbeitung der Operation ausgegeben werden.
- Die REF-Anweisung kann z.B. in einer FOR-NEXT-Anweisung oder zwischen einer CJ-Anweisung (höhere Schrittnummer) und der zugehörigen Pointermarkierung (niedrigere Schrittnummer) eingesetzt werden.

HINWEIS | Der Zustand der Ein- und Ausgänge wird in jedem Programmzyklus aktualisiert.

Beispiel ▾ Einsatz der REF-Anweisung, Eingänge auffrischen**Abb. 6-90:**

Programmierbeispiel zum Einsatz der REF-Anweisung; Eingänge auffrischen

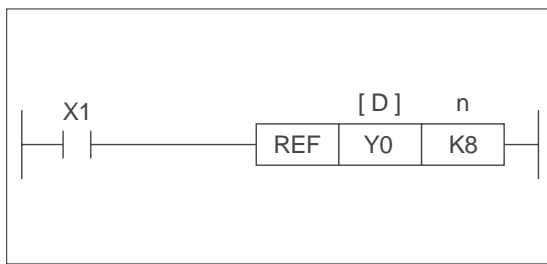
C000106C

8 Adressen, d.h. die Eingänge X0 bis X7, werden aufgefrischt.

Sind die Eingänge ungefähr 10 ms (Verzögerungszeit) vor der Abarbeitung der REF-Anweisung aktiviert, wird das Eingangsimageregister aktualisiert, wenn die REF-Anweisung ausgeführt wird. △

HINWEIS

Die Eingangssignalverzögerungszeit können Sie im Datenregister D8020 und D8021 verändern (weitere Hinweise siehe Abs. 10.2.3).

Beispiel ▾ Einsatz der REF-Anweisung, Ausgänge auffrischen**Abb. 6-91:**

Programmierbeispiel zum Einsatz der REF-Anweisung; Ausgänge auffrischen

C000107C

8 Ausgänge, d.h. die Ausgänge Y0 bis Y7, werden aufgefrischt.

Ist einer der Ausgänge eingeschaltet, wird der zugehörige Ausgang des Ausgangsimageregisters nach der Ausführung der REF-Anweisung eingeschaltet. Der Ausgangskontakt wird nach Ablauf der Antwortzeit aktiviert. Die Antwortzeit ist die physikalisch bedingte Schaltzeit des aktivierten Ausganges. △

6.7.2 Einstellen der Eingangfilter (REFF)

		REFF		FNC 51	
		Einstellen der Eingangfilter			
Operanden	n K, H n = 0 – 60 ms	CPU	FX1S	FX1N	FX2N
					●
Operanden	n K, H n = 0 – 60 ms	Puls-Anweisung (P)	Verarbeitung		Programmschritte
			●	16 Bit	32 Bit

Funktionsweise

Einstellen der Filterzeiten der Eingänge X0 bis X7

Beschreibung

- Durch die Ausführung der Anweisung REFF werden die Zustände der Eingänge X0 bis X7 in das Prozessabbild der Eingänge übernommen und die Eingangfilter auf (n) ms eingestellt.
- (n) kann einen Wert zwischen 0 und 60 annehmen. Die Einstellung von 0 bewirkt eine Filterzeit von 50 µs.

HINWEIS

Die Anweisung muß in jedem Zyklus eingeschaltet werden, da die Filtereinstellung ansonsten wieder auf den Standardwert von 10 ms eingestellt wird.

Beispiel ▽

Ist X10 eingeschaltet, wird der Eingangsstatus der Eingänge X0 bis X7 bei einer Eingangsverzögerung von 1 ms aufgefrischt. Normalerweise beträgt die Eingangsverzögerung 10 ms.

Mit der Anweisung „REFF K20“ wird beim Einschalten der Steuerung die Eingangsverzögerung auf 20 ms eingestellt.

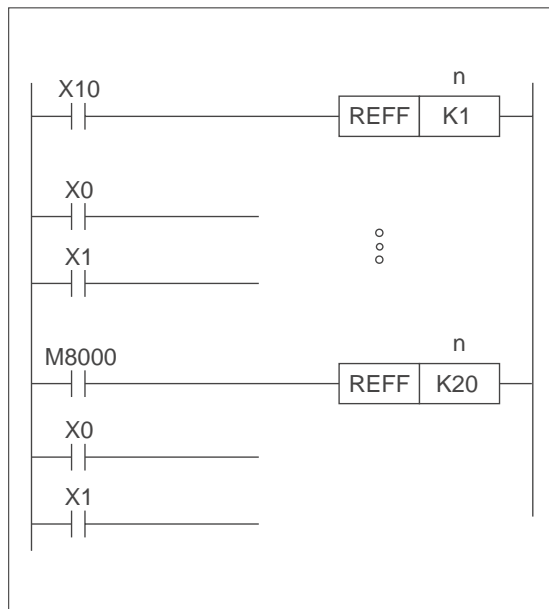


Abb. 6-92:
Programmierbeispiel zur REFF-Anweisung

C000150C



6.7.3 Einlesen einer Matrix (MTR)

				MTR		FNC 52		
				Einlesen einer Matrix				
Operanden				Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					S+, D1+	D2+	n	16 Bit

① Der Operand sollte ein Vielfaches von 10 sein: X0, X10, X20 usw. bis X170

Funktionsweise

Einlesen einer 8 x n-Matrix in die SPS

Beschreibung

- Die Schalter einer 8 x n-Matrix werden in einem Multiplex-Verfahren über 8 Eingänge und n Ausgänge eingelesen.
- Den Eingängen (S+) bis ((S+)+8) werden für jede der n Reihen Merker zugeordnet. Die Merker für die erste Reihe beginnen mit (D2+).
- Jeder Reihe ist ein Ausgang zugeordnet; die erste Reihe wird durch den Ausgang (D1+) angesprochen.

HINWEISE

Zur Ausführung der Anweisung sollte eine Steuerung mit Transistorausgängen genutzt werden.

Die Anweisung kann nur einmal innerhalb eines Programms genutzt werden.

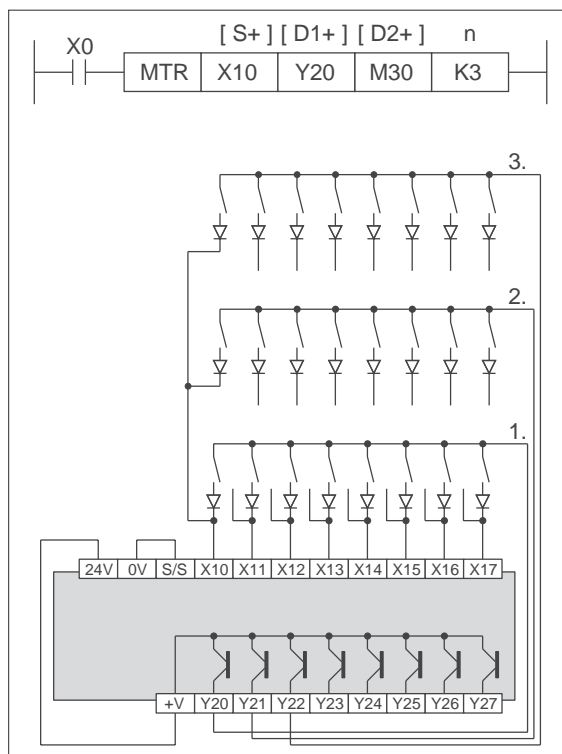
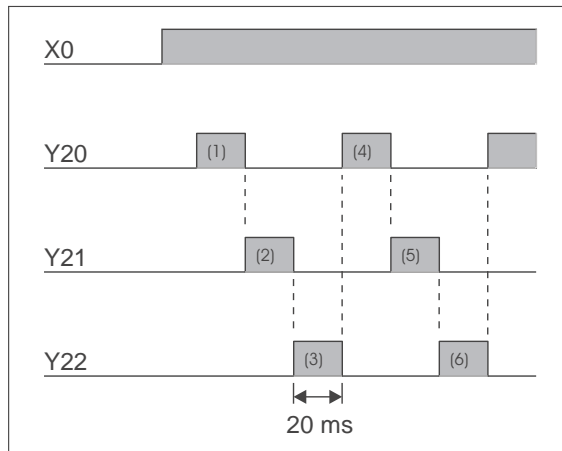


Abb. 6-93:
 Programmier- und Anwendungsbeispiel für eine MTR-Anweisung

C000154C

Beispiel ▾

Im nachfolgenden Beispiel ist zu sehen, daß die 3 Ausgänge Y20, Y21 und Y22 nacheinander eingeschaltet werden. Dieser Vorgang wiederholt sich ständig. Die in der ersten, zweiten und dritten Zeile erfaßten Daten werden kontinuierlich nach M30 bis M37, M40 bis M47 und M50 bis M57 übertragen und gespeichert.

**Abb. 6-94:**

Programmierbeispiel zum Schalten der Ausgänge

C000153C

Die Ein-/Ausgangsbearbeitung für jeden Ausgang wird im Interrupt-Modus in Intervallen von 20 ms ausgeführt, wobei eine Verzögerungszeit der Eingangsfiler von 10 ms berücksichtigt ist.

Mit Hilfe der MTR-Anweisung können 64 Eingangszustände durch den Einsatz von 8 Eingängen und 8 Transistorausgängen erfaßt werden. Alle Eingangsdaten werden innerhalb von 160 ms (20 x 8) eingelesen. Die Eingangsdaten können innerhalb von 80 ms aufgenommen werden, wenn die Eingänge X0 bis X7 eingesetzt werden, da für das Einlesen der Daten pro Zeile nur 10 ms benötigt werden.

M30 bis M37 bleiben unverändert, solange die Eingangsbedingung nicht gesetzt ist. M8029 wird gesetzt, sobald die Matrix gefüllt ist. M8029 wird zurückgesetzt, wenn die Eingangsbedingung ausgeschaltet wird.

△

HINWEISE

Zur Vermeidung von Programmkonflikten sollten als Eingänge möglichst nicht die Adressen X0 bis X7 verwendet werden.

Wurden diese Eingänge doch verwendet, ist pro Ausgang entsprechend der Abb. 6-98 ein Pull-Down-Widerstand zwischenschalten.

Beispiel ▾

Zwischenschaltung von Pull-Down-Widerständen bei Verwendung der Eingänge X0 bis X7.

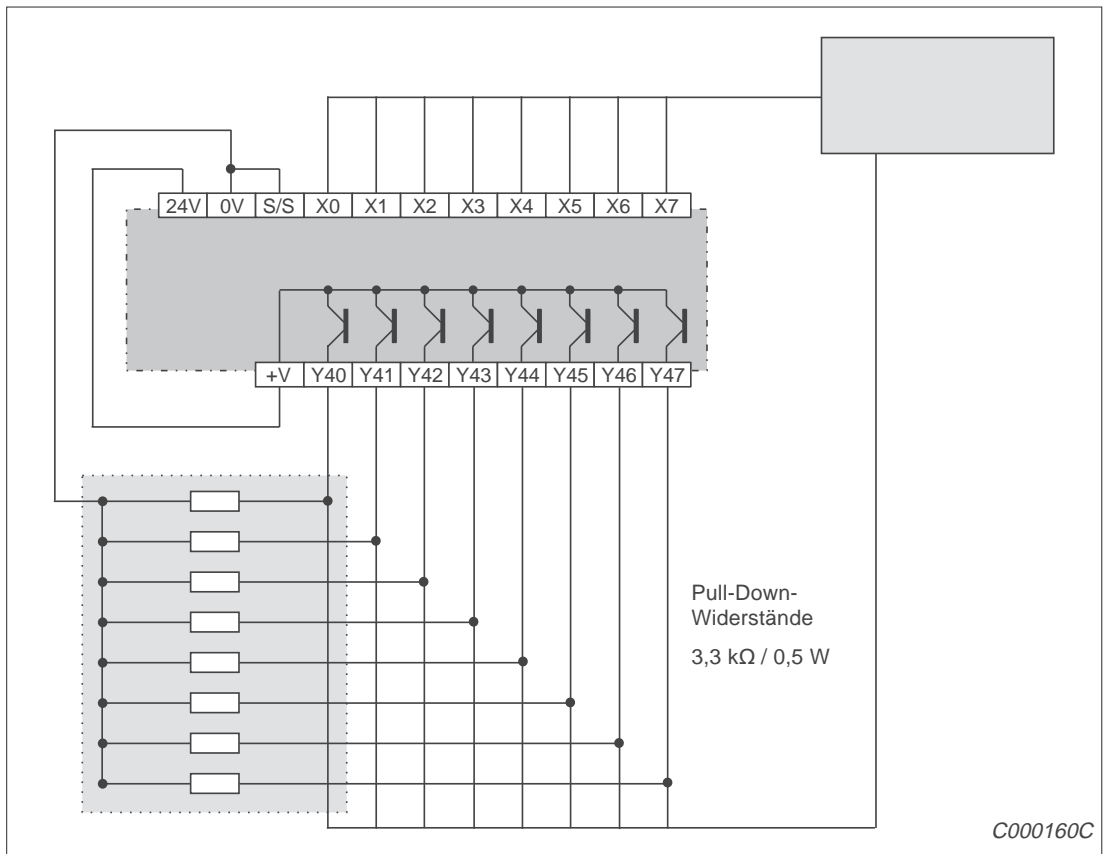


Abb. 6-95: Beispiel zum Einsatz von Pull-Down-Widerständen



6.7.4 Setzen und Rücksetzen durch High-Speed-Counter (DHSCS, DHSCR)

				DHSCS		FNC 53		
				Setzen durch High-Speed-Counter				
Operanden				Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					CPU	FX1S	FX1N	FX2N
S1+	S2+	D+		16 Bit	32 Bit			
K,H,KnX,KnY,KnM,KnS,T,C,D,V,Z			C235 – C254	Y,M,S		●		

				DHSCR		FNC 54		
				Rücksetzen durch High-Speed-Counter				
Operanden				Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					CPU	FX1S	FX1N	FX2N
S1+	S2+	D+		16 Bit	32 Bit			
K,H,KnX,KnY,KnM,KnS,T,C,D,V,Z			C235 – C254	Y,M,S,C235 – C254		●		

Funktionsweise

Setzen und Rücksetzen von Operanden durch High-Speed-Counter. Die Operanden werden ohne eine Programmzykluszeitverzögerung gesetzt bzw. zurückgesetzt.

Beschreibung

- Ein High-Speed-Counter zählt die Statusänderungen an den Zählengängen im Interrupt-Modus. Jedem High-Speed-Counter sind feste Zählengänge mit festen Funktionen zugeordnet.
- Im Kapitel 9.1.9 finden Sie eine ausführliche Leistungsbeschreibung aller vorhandenen High-Speed-Counter und deren Einsatzmöglichkeiten in einem SPS-Programm.
- Mit der DHSCS-Anweisung können Sie Operanden durch High-Speed-Counter setzen. Der in D+ angegebene Operand wird gesetzt, sobald der eingestellte Sollwert des Counters erreicht ist.
- Mit der DHSCR-Anweisung können Sie Operanden durch High-Speed-Counter zurücksetzen. Der in D+ angegebene Operand wird zurückgesetzt, sobald der eingestellte Sollwert des High-Speed-Counters erreicht ist.
- Die Operanden werden ohne eine Programmzykluszeitverzögerung direkt nach Ausführung der Anweisung gesetzt.
- Die Anweisung wird ausgeführt, wenn die Daten in S1+ mit den Daten in S2+ übereinstimmen. Dabei muß die Aktivierung entweder durch einen Impuls an einem Zählengang oder an einem Reset-Eingang erfolgen. Wenn die Aktivierung über einen Reset-Eingang erfolgen soll, muß der Sondermarker M8025 eingeschaltet sein.
- Die Anweisung wird nicht ausgeführt, wenn die Datenübereinstimmung zwischen S1+ und S2+ durch eine indirekte Änderung der Daten in S1+ erreicht wurde. Steht z. B. in S1+ das Datenregister D0 und wird der Datenwert in D0 durch eine MOV-Anweisung verändert, wird die High-Speed-Anweisung nicht ausgeführt.

HINWEIS

In einem SPS-Programm dürfen nicht 6 DHSCS- und DHSCR-Anweisungen eingesetzt werden.

Beispiel ▾ Einsatz der DHSCS-, DHSCR-Anweisungen

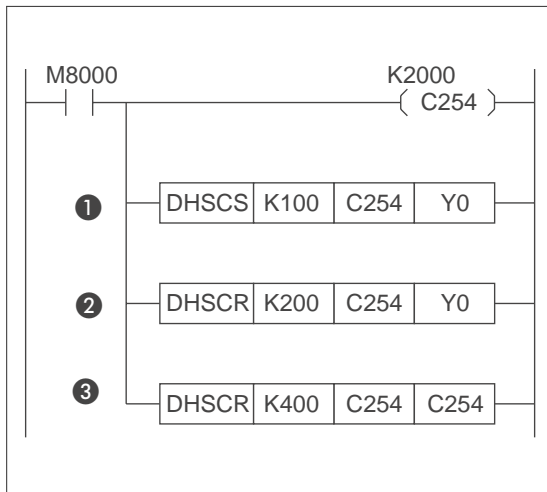


Abb. 6-96:
 Programmierbeispiel zum Einsatz der DHSCS-, DHSCR-Anweisung

C000111C

Der Zählereingang des High-Speed-Counters C254 ist X0 (A-Phase) und X1 (B-Phase). Der Reset-Eingang ist X2, und der Start-Eingang ist X3.

- ① Wenn sich der Istwert des Counters C254 verändert von 99 auf 100 oder von 101 auf 100, wird der Ausgang Y0 sofort gesetzt.
- ② Wenn sich der Istwert des Counters C254 verändert von 199 auf 200 oder von 201 auf 200, wird der Ausgang Y0 sofort zurückgesetzt.
- ③ Wenn sich der Istwert des Counters C254 verändert von 399 auf 400 oder von 401 auf 400, wird der Counter C254 sofort zurückgesetzt.

△

HINWEIS

Der Ausgang wird durch seine physikalische Schaltzeit verzögert. Interne Operanden werden nach der Ausführung der Anweisung im jeweiligen Image-Register eingetragen.

Einsatz von Counter-Interrupt-Pointern

- Counter-Interrupt: I 0 ① 0

① Adresse 1 bis 6

Counter-Interrupts können als Operanden zum Setzen (HSCS, FNC 53) oder Zurücksetzen (HSCR, FNC 54) durch High-Speed-Counter verwendet werden. Zum Ausschalten des Counter-Interrupts schalten Sie den Sondermerker M8059 ein.

Beispiel ▾ Interrupt-Pointer: I030

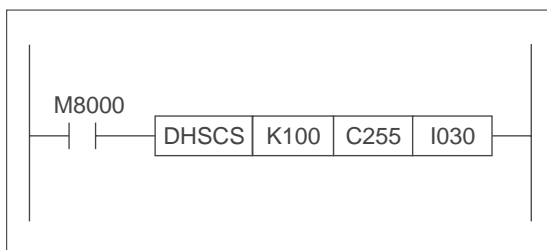


Abb. 6-97:
 Programmierbeispiel zum Einsatz eines Counter-Interrupts

C000333C

Das über den Interrupt-Pointer I030 aufgerufene Interrupt-Programm wird ausgeführt, sobald der Wert des High-Speed-Counters C255 den in K100 angegebenen Wert erreicht. △

HINWEIS

Bitte beachten Sie Absatz 6.7.4 mit den näheren Informationen zum Einsatz der Befehle zum Setzen oder Zurücksetzen durch High-Speed-Counter.

6.7.5 Bereichsvergleich (DHSZ)

				DHSZ		FNC 55		
				Bereichsvergleich				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S1+, S2+	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	C235 bis C255	Y, M, S		16 Bit	32 Bit	DHSZ	17

Funktionsweise ohne Sondermerker (FX1S/FX1N/FX2N)

Bereichsvergleich für High-Speed-Counter im durch (S1+ / S2+) vorgegebenen Bereich

Beschreibung

- Die Anweisung DHSZ vergleicht im Interrupt-Betrieb den Istwert eines High-Speed-Counters mit dem in (S1+ / S2+) vorgegebenen Bereich.
- Der Vergleich findet mit jedem Zählimpuls am angegebenen Zähler (S+) statt.
- Das Ergebnis des Vergleichs wird über die Operanden (D+), ((D+)+1), ((D+)+2) dargestellt.
 - (D+) = $S+ < S1+ / S2+$; Zähler-Istwert unterhalb des Bereichs
 - ((D+)+1) = $S+ = S1+ / S2+$; Zähler-Istwert innerhalb des Bereichs
 - ((D+)+2) = $S+ > S1+ / S2+$; Zähler-Istwert oberhalb des Bereichs
- Da es sich um eine High-Speed-Anweisung handelt, werden Ausgänge, die in (D+) angegeben werden, sofort physikalisch ausgegeben.

HINWEIS

In einem SPS-Programm der FX-Serie dürfen 6 High-Speed-Anweisungen des Typs DHSCS, DHSCR oder DHSZ gleichzeitig aktiv sein.

Beispiel ▾

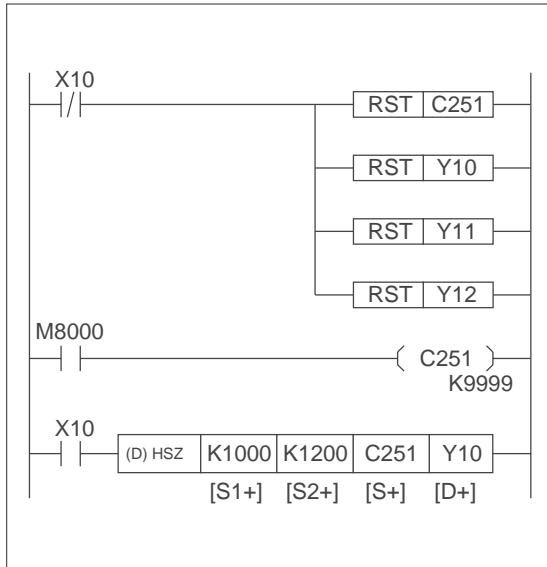


Abb. 6-98:
 Programmierbeispiel zur DHSZ-Anweisung

C000156C

- (S1+): Ende des Eilgangs (Start des Schleichgangs)
- (S2+): Ende des Schleichgangs (Einsetzen der Bremse)
- (S+): Definition des High-Speed-Counters
- (D+): Y10 -> Eilgang
 Y11 -> Schleichgang
 Y12 -> Bremse

Der Zähl- und Vergleichsvorgang sowie die externe Ausgabe werden im Interrupt-Modus ausgeführt.

Beispiel ▴

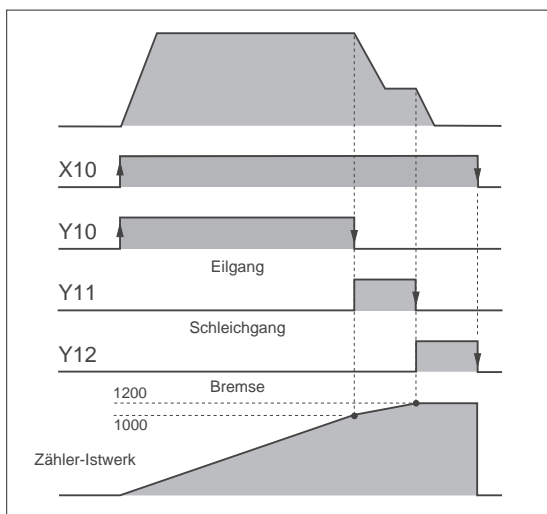


Abb. 6-99:
 Zeitverlauf für das Schalten der Ausgänge Y10, Y11, Y12

C000110C

Entsprechend dem Istwert des Counters C251 werden im Beispiel die Ausgänge Y10 bis Y12 geschaltet.

Ist X10 ausgeschaltet, sind die Ausgänge Y10 bis Y12 ebenfalls ausgeschaltet (aufgrund der RST-Funktion).

Wird ein neuer Counter-Istwert z.B. mit Hilfe der MOV-Anweisung übertragen, bleibt der Ausgangsstatus bis zur Ausführung der nächsten Zähloperation unverändert. ▴

FX2N**Funktionsweise mit Sondermerker M8130 (FX2N)**

Tabellenvergleich für High-Speed-Counter im Bereich ab S1+ und die folgenden n1

Beschreibung

- Die Anweisung DHSZ mit Sondermerker M8130 vergleicht im Interrupt-Betrieb den Istwert eines High-Speed-Counters mit den in einem Tabellenbereich angegebenen Werten.
- Der Operand (D+) ist bei dieser Spezialfunktion durch den Sondermerker M8130 gegeben.
- Die Tabellenlänge wird über einen Konstantenwert (K, H) angegeben. Die Maximallänge beträgt 128 Einträge. Je Eintrag der Tabelle werden 4 Datenregister belegt. Zu jedem Eintrag müssen die folgenden Angaben gespeichert werden:
 - der zu vergleichende Wert,
 - der zu adressierende Ausgang (hexadezimal),
 - die Setz- bzw. Rücksetz-Anweisung.
- Der Vergleich findet mit jedem Zählimpuls am angegebenen Zähler (S+) statt.

HINWEISE

In einem SPS-Programm für die FX-Serie dürfen nur 6 High-Speed-Anweisungen des Typs DHSCS, DHSCR oder DHSZ gleichzeitig aktiv sein.

Es kann immer nur eine DHSZ-Anweisung den Sondermerker M8130 verwenden.

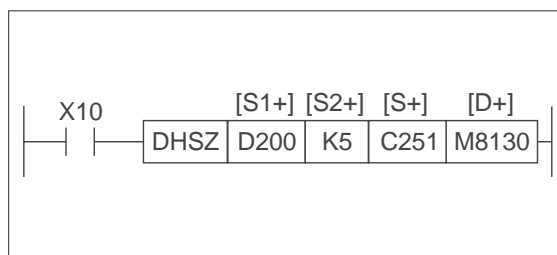
Es muß eine korrekte HSC-Anweisung gesetzt sein.

Die DHSZ-Anweisung mit Sondermerker M8130 wird erstmalig nach der ersten END-Anweisung ausgeführt. Dies erlaubt der Steuerung, die interne Vergleichstabelle aufzubauen.

Der Vergleich in der Tabelle erfolgt immer der Reihenfolge nach. Aus diesem Grund müssen die Vergleichswerte immer in aufsteigender oder absteigender Folge sortiert sein.

Beispiel ▾

Einsatz der DHSZ-Anweisung mit Sondermerker M8130

**Abb. 6-100:**

Programmierbeispiel zur DHSZ-Anweisung mit Sondermerker M8130

C000322C

Nach Setzen des Eingangs X10 erfolgt der Tabellenvergleich ab Datenregister D200 in 5 Einträgen der Datentabelle mit dem Wert in C251.

Die Tabelle hat die folgende Form (K1 = Setzen, K0 = Rücksetzen):

Eintrag Nr.	Vergleichswert		Ausgabe-Anweisung	Setzen/Zurücksetzen
	unterer	oberer		
0	D200	D201	D202	D203
	K123		H10 (=Y10)	K1
1	D204	D205	D206	D207
	K234		H10	K0
2	D208	D209	D210	D211
	K345		H23 (=Y23)	K1
3	D212	D213	D214	D215
	K456		H23	K0
4	D216	D217	D218	D219
	K567		H23	K1

Tab. 6-22:
Vergleichstabelle, Startadresse D200, Länge K5

Der Vergleichswert ist im 32-Bit-Format (Doppelwort) gespeichert. Der Ausgang ist im Hexadezimal-Format angegeben.

Beispiel  

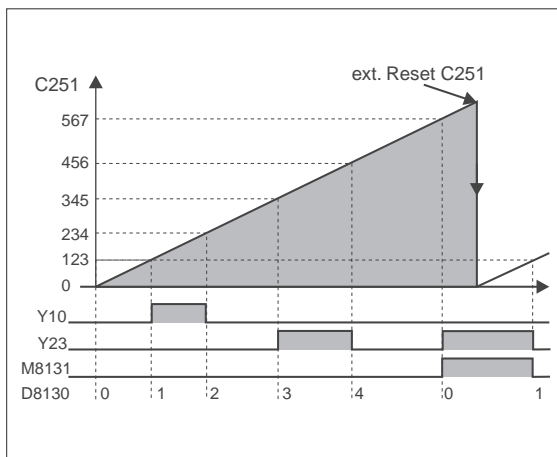


Abb. 6-101:
Zeitverlauf für das Schalten der Ausgänge Y10, Y23

C000323C

Wird die DHSZ-Anweisung mit dem Sondermarker M8130 gesetzt, wird Datenregister D8130 als Zähler der Eintragsnummern definiert. Nach jedem Vergleich springt D8130 zur nächsten Eintragsnummer.

Sind alle Einträge der Tabelle abgearbeitet, wird das Vorgangsende-Flag M8131 eingeschaltet und D8130 durch einen externen oder durch einen Programm-Impuls zurückgesetzt. D8130 startet erneut mit der Zählung, wenn Flag M8131 zurückgesetzt wird.



FX2N

Funktionsweise mit Sondermerker M8132 (FX2N)

Tabellenvergleich für High-Speed-Counter im Bereich ab S1+ und die folgenden n1 mit ergebnisabhängiger Frequenzsteuerung in der DPLSY-Anweisung.

Beschreibung

- Die Anweisung DHSZ mit Sondermerker M8132 vergleicht im Interrupt-Betrieb den Istwert eines High-Speed-Counters mit den in einem Tabellenbereich angegebenen Werten. Bei Übereinstimmung wird ein in der Tabelle angegebener Wert an die folgende DPLSY-Anweisung (FNC 57) zur Frequenzsteuerung ausgegeben.
- Der Operand (D+) ist bei dieser Spezialfunktion durch den Sondermerker M8132 gegeben.
- Die Tabellenlänge wird über einen Konstantenwert (K, H) angegeben. Die Maximallänge beträgt 128 Einträge. Je Eintrag der Tabelle werden 4 Datenregister belegt. Zu jedem Eintrag müssen die folgenden Angaben gespeichert werden:
 - der zu vergleichende Wert (32-Bit-Format),
 - der zu adressierende Ausgang (32-Bit-Format).
- Der Vergleich findet mit jedem Zählimpuls am angegebenen Zähler (S+) statt.

HINWEISE

In einem SPS-Programm für die FX-Serie dürfen nur 6 High-Speed-Anweisungen des Typs DHSCS, DHSCR oder DHSZ gleichzeitig aktiv sein.

Es kann immer nur eine DHSZ-Anweisung den Sondermerker M8132 verwenden.

Es muß eine korrekte HSC-Anweisung gesetzt sein.

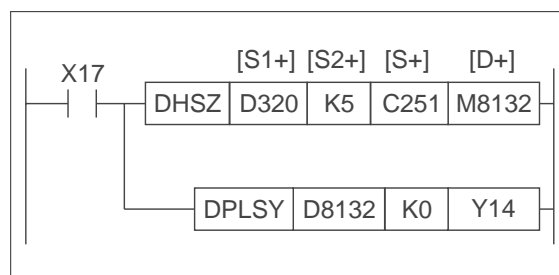
Die DHSZ-Anweisung mit Sondermerker M8132 wird erstmalig nach der ersten END-Anweisung ausgeführt. Dies erlaubt der Steuerung, die interne Vergleichstabelle aufzubauen.

Der Vergleich in der Tabelle erfolgt immer der Reihenfolge nach. Aus diesem Grund müssen die Vergleichswerte immer in aufsteigender oder absteigender Folge sortiert sein.

Der letzte Eintrag in der Tabelle sollte auf (K0, K0) gesetzt werden, um sicherzustellen, daß die Impulsausgabe gestoppt wird und D8131 nicht zum Tabellenkopf zurückspringt. Register D8134 und D8135 erhalten den Wert K0 und geben das Ende der Tabelle an.

Beispiel ▾

Einsatz der DHSZ-Anweisung mit Sondermerker M8132

**Abb. 6-102:**

Programmierbeispiel zur DHSZ-Anweisung mit Sondermerker M8132

C000324C

Nach Setzen des Eingangs X10 erfolgt der Tabellenvergleich ab Datenregister D200 in 5 Einträgen der Datentabelle mit dem Wert in C251.

Die Tabelle hat die folgende Form:

Eintrag Nr.	Vergleichswert		Ausgangsfrequenz	
	unterer	oberer	unterer	oberer
0	D320	D321	D322	D323
	K20		K300	
1	D324	D325	D326	D327
	K600		K500	
2	D328	D329	D330	D331
	K700		K200	
3	D332	D333	D334	D335
	K800		K100	
4	D336	D337	D338	D339
	K0		K0	

Tab. 6-23:
Vergleichstabelle, Startadresse D320, Länge K5

Der Vergleichswert ist im 32-Bit-Format (Doppelwort) gespeichert, der Ausgangswert gibt die Frequenz an, die gesetzt bleibt, bis das Vergleichsergebn übereinstimmt.

Beispiel  

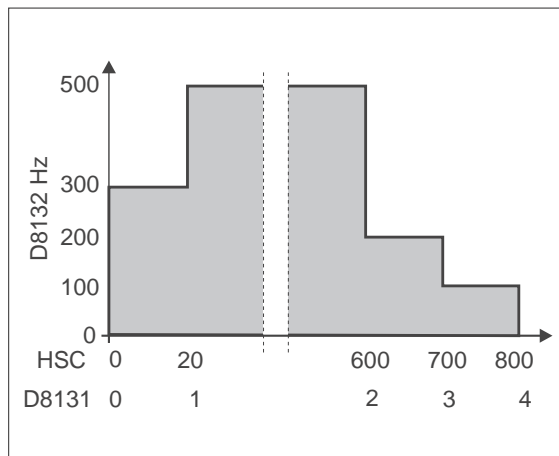


Abb. 6-103:
Frequenzverlauf bei Anwendung der DHSZ-Anweisung mit Sondermerker M8132

C000325C

Wird die DHSZ-Anweisung mit dem Sondermerker M8132 gesetzt, wird Datenregister D8131 als Zähler der Eintragsnummern definiert. Nach jedem Vergleich springt D8131 zur nächsten Eintragsnummer.

D8132 erhält den jeweiligen Frequenzwert des Tabelleneintrags zur Verwendung in der PLSY-Anweisung. Zur Abarbeitung tragen die Datenregister D8134 und D8135 den zu vergleichenden Wert.

Sind alle Einträge der Tabelle abgearbeitet, wird Vorgangsende-Flag M8133 eingeschaltet und D8131 durch einen externen oder durch einen Programm-Impuls zurückgesetzt. D8131 startet erneut mit der Zählung, wenn Flag M8133 zurückgesetzt wird.

Bei Zurücksetzen der DHSZ-Anweisung werden alle Werte, einschließlich der Frequenzangabe, zurückgesetzt.



6.7.6 Geschwindigkeitserkennung (SPD)

				SPD		FNC 56		
				Geschwindigkeitserkennung				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X0 bis X5	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	T, C, D			16 Bit	32 Bit	SPD
					●			

Funktionsweise

Erfassung der Anzahl der Impulse innerhalb einer vorgewählten Zeit

Beschreibung

- Die Impulse an (S1+) werden für (S2+) in ms gezählt, und das Ergebnis wird in (D+) abgelegt.
- Es werden die Operanden (D+), ((D+)+1) und ((D+)+2) belegt.
 - (D+): Summe der Impulse nach Ablauf der Zeit
 - ((D+)+1): aktueller Zählwert innerhalb des Zeitintervalls
 - ((D+)+2): verbleibende zu zählende Zeit

HINWEISE

Nach Ablauf der Zeit wird der Inhalt von ((D+)+1) nach (D+) übertragen, und ((D+)+1) wird zurückgesetzt.

Die in der Anweisung verwendeten High-Speed-Eingänge dürfen nicht in anderen High-Speed-Operationen verwendet werden.

Für jeden High-Speed-Eingang kann max. eine SPD-Anweisung gegeben werden.

Beispiel ▾

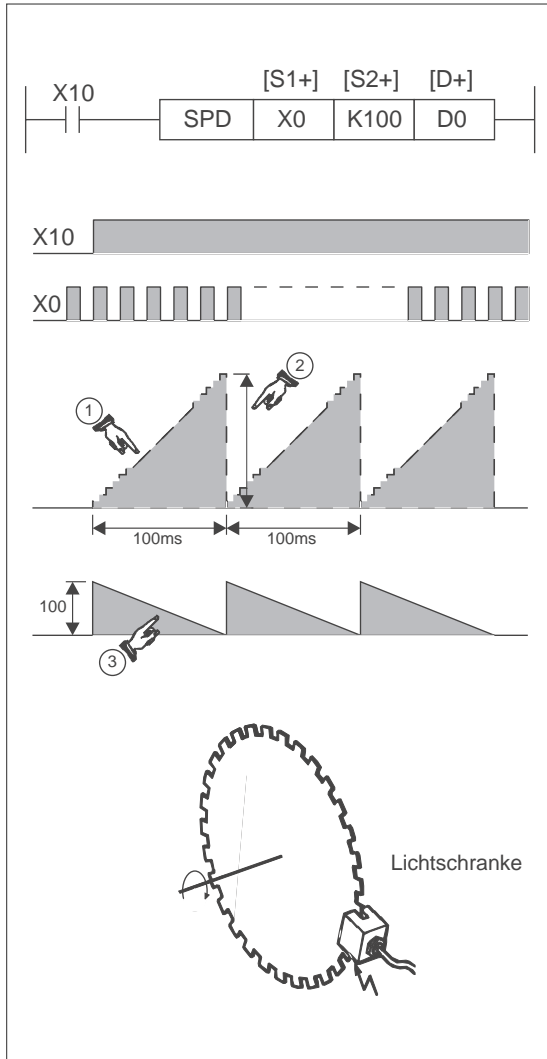


Abb. 6-104:
 Programmierbeispiel zur SPD-Anweisung

C000158C

- ①: Istwert (D1)
- ②: Zähleristwert (D0)
- ③: Verbleibende Zeit (D2)

Im Beispiel zählt D1 die Anzahl der Einschaltvorgänge von X0. Nach 100 ms wird das Zählergebnis in D0 gespeichert.

D1 wird zurückgesetzt und beginnt erneut mit dem Zählen der Einschaltvorgänge von X0.

In D2 wird die jeweils verbleibende Zeit gemessen.

Mit diesem Wert kann die Drehzahl eines Antriebs ermittelt werden.

$$N = \frac{60 \times D0}{n \times t} \times 10^3 \text{ (U/min.)}$$

n: Impuls/Umdrehung

N: Geschwindigkeit

t: Intervalle (ms), die in S2+ angegeben sind



6.7.7 Ausgabe einer definierten Anzahl von Impulsen (PLSY, DPLSY)

		PLSY		FNC 57			
		Ausgabe einer definierten Anzahl von Impulsen					
		CPU	FX1S	FX1N	FX2N		
			●	●	●		
Operanden	S1+, S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	Nur Y0 oder Y1		16 Bit	32 Bit	PLSY	7
				●	●	DPLSY	13

Funktionsweise

Definierte Anzahl von Impulsen mit einer festgelegten Frequenz und einem festen Pulsweitenverhältnis von 50 : 50 über einen Ausgang ausgeben

Beschreibung

- Die Anweisung erzeugt eine definierte Anzahl von Impulsen.
- In (S1+) wird die Frequenz festgelegt: 1 bis 32767 Hz (FX1S/FX1N, PLSY)
1 bis 100 kHz (FX1S/FX1N, DPLSY)
2 bis 20 kHz (FX2N)
- In (S2+) wird die Anzahl der zu erzeugenden Impulse angegeben. Dabei dürfen folgende Wertebereiche nicht überschritten werden.
16-Bit-Anweisung: 1 bis 32 767 Impulse
32-Bit-Anweisung: 1 bis 2 147 483 647 Impulse
Wird der Wert 0 eingegeben, werden fortlaufend Impulse erzeugt.
- In (D+) wird die Adresse des Ausgangs festgelegt.
- Das Verhältnis zwischen EIN- und AUS-Zustand beträgt: 50 % EIN, 50 % AUS
Die EIN- und AUS-Zustände werden direkt im Interrupt-Modus ausgegeben.
- Bei Verwendung der DPLSY-Anweisung wird die Anzahl der Impulse in zwei aufeinanderfolgenden Datenregistern angegeben.
- Ist die gewünschte Anzahl von Impulsen erzeugt, wird der Sondermarker M8029 (Anweisung vollständig abgearbeitet) gesetzt. M8029 wird zurückgesetzt, wenn die PLSY-Anweisung deaktiviert wird.

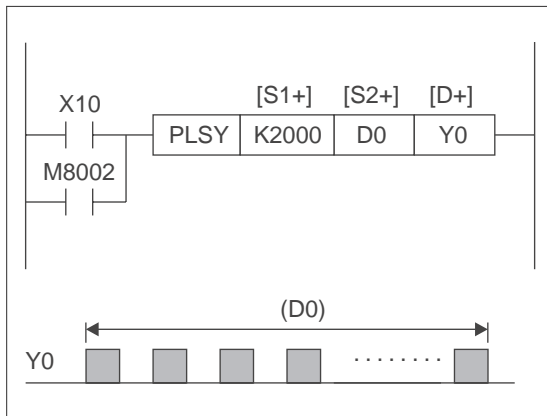
Die Daten in (S1+) (Frequenz) können während der Ausführung der Anweisung verändert werden. Veränderte Daten in (S2+) (Anzahl der Impulse) kommen jedoch erst dann zum Einsatz, wenn die Anweisung abgearbeitet ist.

HINWEISE

- | Die Anweisung kann nur einmal im Programm eingesetzt werden.
- | Die Impulse können nur über die Ausgänge Y0 und Y1 ausgegeben werden.
- | Die Steuerung sollte über Transistorausgänge verfügen.
- | Der Laststrom am Ausgang Y0 sollte mindestens 200 mA betragen, damit eine einwandfreie Funktion auch bei max. Frequenz gewährleistet ist.
- | Einsatz mit DHSZ-Anweisung siehe Seite 6-102.

Beispiel ▾

Einsatz der PLSY-Anweisung

**Abb. 6-105:**

Programmierbeispiel zum Einsatz der PLSY-Anweisung

C000105C

Ist X10 eingeschaltet, werden Impulse mit einer Frequenz von 2000 Hz erzeugt. Es werden immer soviele Impulse erzeugt, wie im Datenregister D0 angegeben ist.

Die Erzeugung der Impulse wird eingestellt, wenn X10 ausgeschaltet wird. Wird X10 wieder eingeschaltet, beginnt die Operation erneut. Ist X10 nicht gesetzt, wird Y0 ausgeschaltet.

△

HINWEISE

Bei der FX2N-Serie zeigt das Doppelwort in D8140 und D8141 die aktuellen Pulse von Y0 an. Das Doppelwort D8142 und D8143 zeigt die Pulse von Y1 an.

In D8136 und D8137 wird bei der FX2N-Serie die Summe der Pulse hinterlegt.

6.7.8 Impulsausgabe mit Modulation der Impulsweite (PWM)

			PWM		FNC 58		
			Impulsausgabe mit Modulation der Impulsweite				
Operanden S1+, S2+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z (S1 ≤ S2)			D+ Nur Y0 oder Y1			Puls-Anweisung (P)	
						Verarbeitung	
						CPU	
						FX1S FX1N FX2N	
						● ● ●	
						16 Bit 32 Bit	
						●	
						PWM 7	

Funktionsweise

Fortlaufend Impulse mit festgelegter Impulsweite und Periodendauer (bzw. Frequenz) über einen Ausgang ausgeben

Beschreibung

- Die Anweisung erzeugt fortlaufend Impulse. Das Verhältnis zwischen Impulsweite t und Periodendauer T0 wird überwacht.
 t: Impulsweite [ms]
 T0: Periodendauer [ms]
 Frequenz f: 1/T0 [kHz]
- In (S1+) wird die Impulsweite im Bereich von t: 1 bis 32 767 ms festgelegt. Die Impulsweite muss im Bereich von $1 \leq t \leq T0$ liegen.
- In (S2+) wird die Periodendauer im Bereich von T0: 1 bis 32 767 ms festgelegt.
- In (D+) wird die Adresse des Ausgangs angegeben.
- Die Überwachung des EIN-und AUS-Zustands des Ausgangs wird im Interrupt-Modus ausgeführt.

HINWEISE

- | Die Anweisung kann nur einmal in einem Programm eingesetzt werden.
- | Es können nur die Ausgänge Y0 und Y1 angesprochen werden.
- | Die Steuerung sollte über Transistorausgänge verfügen, um einen Kontaktverschleiß zu vermeiden.
- | Der Laststrom der Ausgänge sollte mindestens 200 mA betragen, um eine einwandfreie Funktion auch bei minimaler Periodendauer von T0: 1 ms zu gewährleisten.

Fehlerquelle

Ist der festgelegte Wert für die Impulsweite t in (S1+) größer als der festgelegte Wert für T0 in (S2+), tritt ein Programmablauffehler auf.

Beispiel ▾

Einsatz der PWM-Anweisung

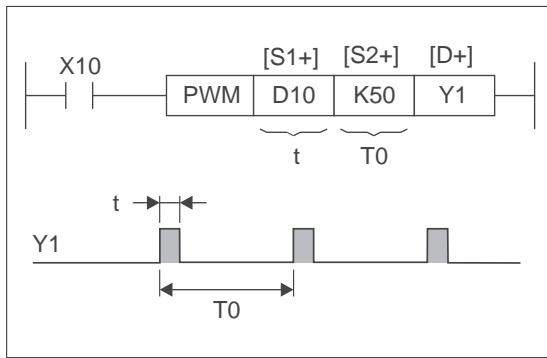


Abb. 6-106:

Programmierbeispiel zum Einsatz der PWM-Anweisung

C000113C

Durch Ändern der Daten im Register D10 in einem Bereich von 0 bis 50 kann die relative Pulsweite T0 von 0 % bis 100 % variiert werden. Wird der Wert D10 auf 0 gesetzt, wird kein Impuls ausgegeben. Wird der Wert von D10 auf 50 geändert, so ist Y1 für den ganzen Zyklus gesetzt.

Y1 wird ausgeschaltet, wenn X10 ausgeschaltet ist.

△

6.7.9 Ausgabe einer bestimmten Anzahl von Impulsen (PLSR)

		PLSR		FNC 59				
		Ausgabe einer bestimmten Anzahl von Impulsen						
Operanden		S1+, S2+, S3+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z (S1 ≤ S2)	D+ Nur Y0 oder Y1	Puls-Anweisung (P)	CPU	FX1S	FX1N	FX2N
					●	●	●	
					Verarbeitung		Programmschritte	
					16 Bit	32 Bit	PLSR	9
					●	●	DPLSR	17

Funktion

Eine festgelegte Anzahl von Impulsen mit festgelegter Frequenz an einen Ausgang ausgeben.

Beschreibung

- Die PLSR-Anweisung erzeugt an einem Ausgang eine angegebene Anzahl von Impulsen (S2+) mit angegebener Frequenz (S1+). Die Frequenz wird in einer angegebenen Zeit (S3+) in 10 Schritten hoch- und wieder heruntergefahren.
- Die Ausgabefrequenz kann bei einer SPS der FX2N-Serie zwischen 10 und 20.000 Hz liegen. Bei Steuerungen der FX1S- oder FX1N-Serie können Frequenzen von 10 bis 100.000 Hz ausgegeben werden. Die angegebene Frequenz sollte durch 10 teilbar sein. Ist die angegebene Frequenz nicht durch 10 teilbar, wird die Frequenz auf einen entsprechenden Wert aufgerundet.
- Die Schrittweite der Rampen beträgt 1/10-tel der angegebenen Ausgabefrequenz (bei der Verwendung von Schrittmotoren zu berücksichtigen).
- Die Anzahl der Ausgangsimpulse kann bei Verwendung der 16-Bit-Anweisung zwischen 110 und 32.767 Impulsen liegen.
- Die Anzahl der Ausgangsimpulse kann bei Verwendung der 32-Bit-Anweisung zwischen 110 und 2.147.483.647 Impulsen liegen.
- Bei einer Angabe unter 110 Impulsen kann eine korrekte Impulsausgabe nicht garantiert werden.
- Die Anstiegszeit der Rampe muß den anschließend beschriebenen Grenzwerten entsprechen.
- Als Ausgänge können nur Y0 und Y1 programmiert werden.

HINWEISE

In einem Programm können zur selben Zeit zwei PLSR-Anweisungen verwendet werden, um Impulse an Y0 und Y1 auszugeben. Möglich ist auch die Verwendung von einer PLSY-Anweisung (FNC57) und einer PLSR-Anweisung (FNC59) in einem Zyklus, um Impulse an Y0 und Y1 auszugeben. Eine mehrfache Verwendung kann durch Unterprogramme oder ähnliche Verfahren realisiert werden.

Wenn die Anzahl der angegebenen Impulse nicht ausreicht, um die angegebene Frequenz zu erreichen, wird die Frequenz abgeschnitten.

Der Sondermerker M8029 wird nach Ausgabe der angegebenen Anzahl von Impulsen gesetzt. Das Rücksetzen des Merkers erfolgt mit Rücksetzen der Ausführungsbedingung der PLSR-Anweisung.

Begrenzung der Rampenanstiegszeit

Die Anstiegszeit (S3+) ist auf 5.000 ms begrenzt. Die Grenzwerte der Anstiegszeit in Abhängigkeit der Frequenz und Anzahl der Ausgabeimpulse werden wie folgt berechnet:

- Der Wert in (S3+) muß mindestens 10 mal größer als die Programmzykluszeit sein (D8012). Bei Unterschreitung des Wertes erfolgt der Rampenanstieg in ungeraden Schritten.
- Der Minimalwert für (S3+) errechnet sich nach folgender Gleichung:

$$(S3+) \geq (9.000 / (S1+)) \times 5$$
- Der Maximalwert für (S3+) errechnet sich nach folgender Gleichung:

$$(S3+) \leq ((S2+) / (S1+)) \times 818$$
- Wenn die Parameter außerhalb der berechneten Grenzen liegen, ist der Wert von (S1+) zu verkleinern.
- Der Anstieg der Ausgabefrequenz erfolgt in 10 Schritten.

Hinweise

Die maximale Ausgabefrequenz und die Frequenzschrittweite der Rampe sind innerhalb des Bereichs 2 bis 20.000 Hz begrenzt.

Nach dem Rücksetzen der Ausführungsbedingung der PLSR-Anweisung werden die angesprochenen Ausgänge zurückgesetzt. Bei erneutem Setzen der Ausführungsbedingung beginnt die Verarbeitung der Anweisung von vorne.

Werden während der Verarbeitung die Operanden verändert, bleibt das Ausgabeprofil der Anweisung erhalten. Die Operandenänderungen werden bei der nächsten Verarbeitung der Anweisung wirksam.

Beispiel ▾

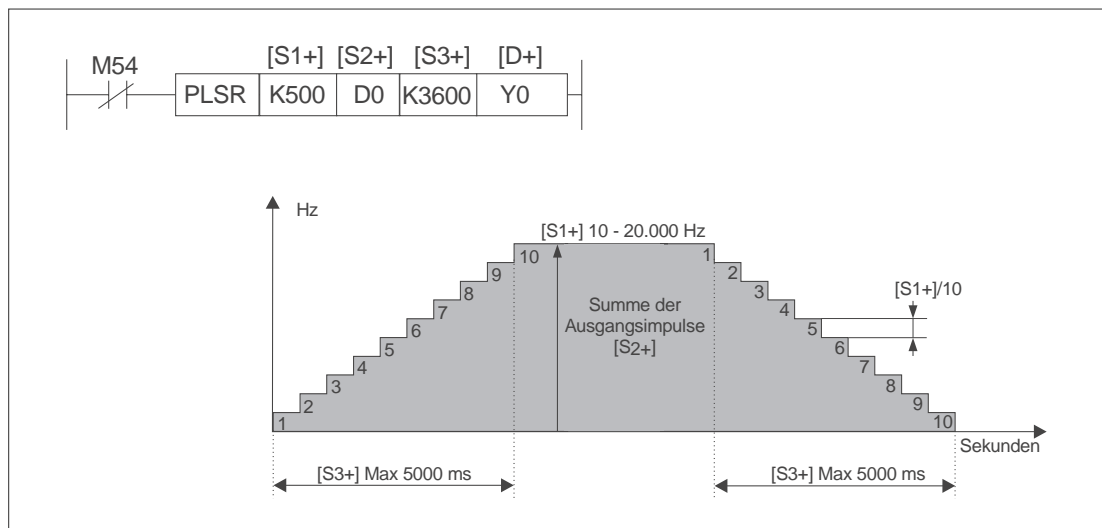


Abb. 6-107: Programmierbeispiel zum Einsatz der PLSR-Anweisung

Mit Rücksetzen des Merkers M54 wird die in D0 (S2+) angegebene Anzahl von Impulsen an Y0 (D+) ausgegeben.

Die Ausgabefrequenz beträgt 500 Hz (S1+).

Der Frequenzanstieg auf 500 Hz (S1+) und die Frequenzabsenkung auf 0 Hz erfolgt in jeweils 3600 ms (S3+) in Schritten zu 50 Hz (S1+ / 10).



6.8 Anwendungsbezogene Anweisungen

Übersicht der Anweisungen FNC 60 bis 69

Symbol	FNC	Bedeutung	Abschnitt
IST	60	Schrittstatus initialisieren	6.8.1
SER	61	Suchanweisung	6.8.2
ABSD	62	Absoluter Counter-Vergleich	6.8.3
INCD	63	Inkrementaler Counter-Vergleich	6.8.4
TTMR	64	Teaching-Timer	6.8.5
STM	65	Sonder-Timer	6.8.6
ALT	66	Flip-Flop-Funktion	6.8.7
RAMP	67	Rampenfunktion	6.8.8
ROTC	68	Rundtisch-Positionierung	6.8.9
SORT	69	Sortieranweisung	6.8.10

Tab. 6-24: Übersicht der Anweisungen FNC 60 bis 69

6.8.1 Schrittstatus initialisieren (IST)

			IST		FNC 60		
			Schrittstatus initialisieren				
			CPU	FX1S	FX1N	FX2N	
				●	●	●	
Operanden	S+	D1+, D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S	FX1S: S20-S127; FX1N: S20-S999 FX2N: S20-S999 (D1+ < D2+)		16 Bit	32 Bit	IST	7
				●			

Funktionsweise

Zuweisen von Sonderfunktionen und Reservieren von Schrittstatusoperanden für eine Schrittsteuerung. Mit der IST-Anweisung können verschiedene Schrittketten über ein Schalt-pult gekoppelt werden. Es können Schrittketten für Automatikbetrieb, Handbetrieb und Nullpunktfahrt initialisiert werden.

Beschreibung

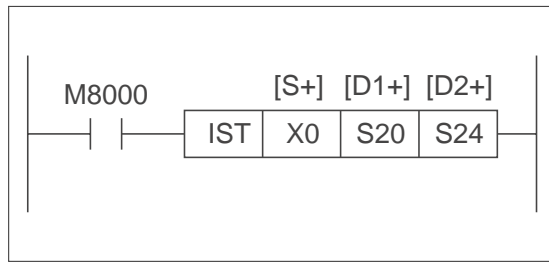
- Die Schrittstatusoperanden S0 bis S2 werden für die Initialisierung der Schrittketten
 - Handbetrieb,
 - Automatikbetrieb und
 - Nullpunktfahrt reserviert.
- Die Schrittstatusoperanden S0 bis S2 benötigen keine SET-Anweisung.
- Die Schrittstatusoperanden S3 bis S9 bleiben frei.
- Die Schrittstatusoperanden S10 bis S19 werden für die Schrittkette der Nullpunktfahrt reserviert.
- Für die Programmierung der übrigen Schrittketten stehen also die verbleibenden Schrittstatusoperanden S20 bis S127 (S999) zur Verfügung.
- Die Sondermerker M8040 bis M8042 und M8047 werden gesteuert.
- In (D1+) und (D2+) wird der Schrittstatusoperandenbereich für die Schrittkette des Automatikbetriebs festgelegt. Dabei muß gelten (D1+) < (D2+).
- In (S+) wird der Eingangsbereich der Steuereingänge festgelegt. Als Steuereingänge können die Operanden X, Y oder M verwendet werden. Es wird die Startadresse des Operandenbereichs angegeben.

HINWEIS

| Die IST-Anweisung darf nur einmal in einem Programm verwendet werden.

Beispiel ▾

Einsatz der IST-Anweisung

**Abb. 6-108:**

Programmierbeispiel zum Einsatz der IST-Anweisung

C000157C

Die Eingänge erhalten in diesem Beispiel folgende Sonderfunktionen:

- X0: Handbetrieb
- X1: Rückkehr in die Ausgangsposition
- X2: Schrittbetrieb
- X3: Einzeloperation
- X4: Automatikbetrieb
- X5: Taster zur Rückkehr in die Ausgangsposition
- X6: Taster zum Starten des Automatikbetriebs
- X7: Taster zum Stoppen des Automatikbetriebs

△

HINWEIS

Diese Steuereingänge dienen der direkten Steuerung des Prozesses. Mit den Steuereingängen werden die Betriebsarten angewählt.

Die folgenden Sondermerker werden durch die IST-Anweisung beeinflusst bzw. steuern die Verarbeitung der Schrittketten:

- M8040: Weiterschalten in einen anderen Schritt möglich
- M8041: Beginn der Weiterschaltung
- M8042: Startimpuls
- M8043: Nullpunkt erreicht
- M8044: Nullpunkt erkannt

Beispiel ▾

Ein Containerfahrzeug für Schüttguttransporte soll im Automatikbetrieb ständig be- und entladen werden.

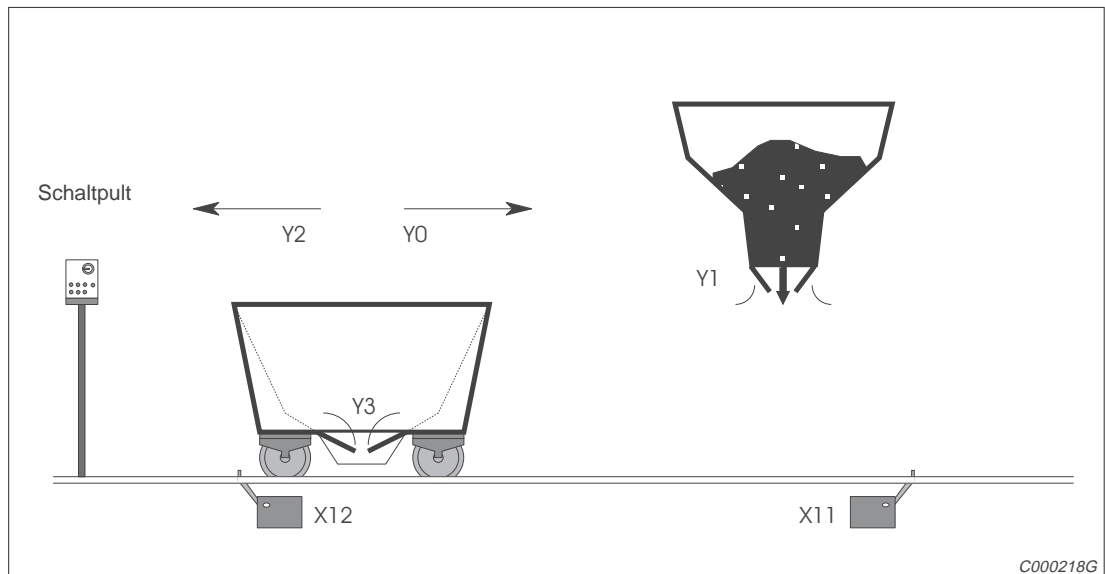


Abb. 6-109: Beispiel, Be- und Entladekontrolle eines Containerfahrzeuges mit Hilfe einer IST-Anweisung

Beschreibung der Signalgeber und Stellglieder

- Endschalter
links: X12
rechts: X11
- Fahrrichtung des Wagens
links: Y2
rechts: Y0
- Silolade öffnet sich für 7 Sekunden: Y1
- Entladeklappe des Wagens öffnet sich für 5 Sekunden: Y3

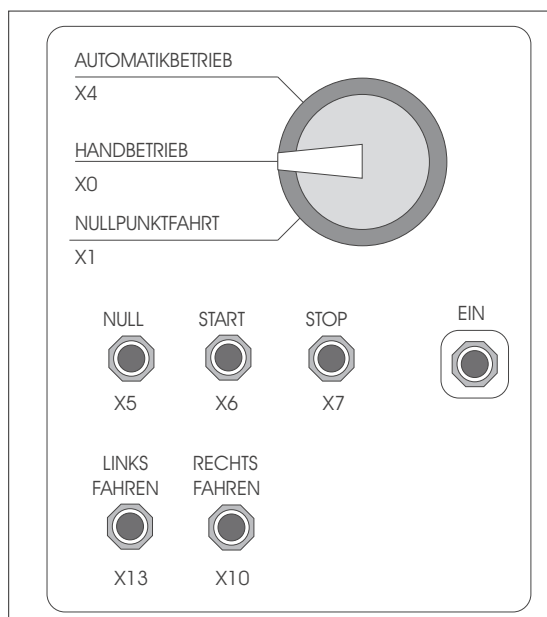


Abb. 6-110: Muster eines Schaltpults

C000217G

Beispiel

Durch Einsatz der IST-Anweisung vorgegebene Sonderfunktionen:

- X0: Handbetrieb
- X1: Rückkehr in die Ausgangsposition
- X2: Schrittbetrieb
- X3: Einzeloperation
- X4: Automatikbetrieb
- X5: Taster zur Rückkehr in die Ausgangsposition
- X6: Taster zum Starten des Automatikbetriebs
- X7: Taster zum Stoppen des Automatikbetriebs

Funktionsbeschreibung

Im Automatikbetrieb soll das Containerfahrzeug ständig zwischen der Be- und Entladestation pendeln. Der Automatikbetrieb wird gestartet, wenn X4 eingeschaltet ist und X6 betätigt wird. Als weitere Einschaltbedingung gilt, daß das Fahrzeug vorher in die Ausgangsposition (Nullpunkt) gefahren wurde (M8043 und M8044 aktiviert). Der Automatikbetrieb wird mit X7 beendet.

Im Handbetrieb kann das Fahrzeug in beiden Bewegungsrichtungen verfahren werden. Hierfür muß X0 eingeschaltet sein. Das Fahrzeug fährt durch Betätigen des Tasters X13 bzw. X10 nach links bzw. nach rechts.

Die Nullpunktfahrt wird durch einen eingeschalteten X1 und einer Betätigung von X5 gestartet. Dadurch wird das Fahrzeug aus jeder beliebigen Position wieder in die Ausgangsposition zurückgefahren.

Die Ausgangsposition ist erreicht, wenn sich das Fahrzeug an der Endschalterposition X12 befindet und entleert ist.



Beispiel

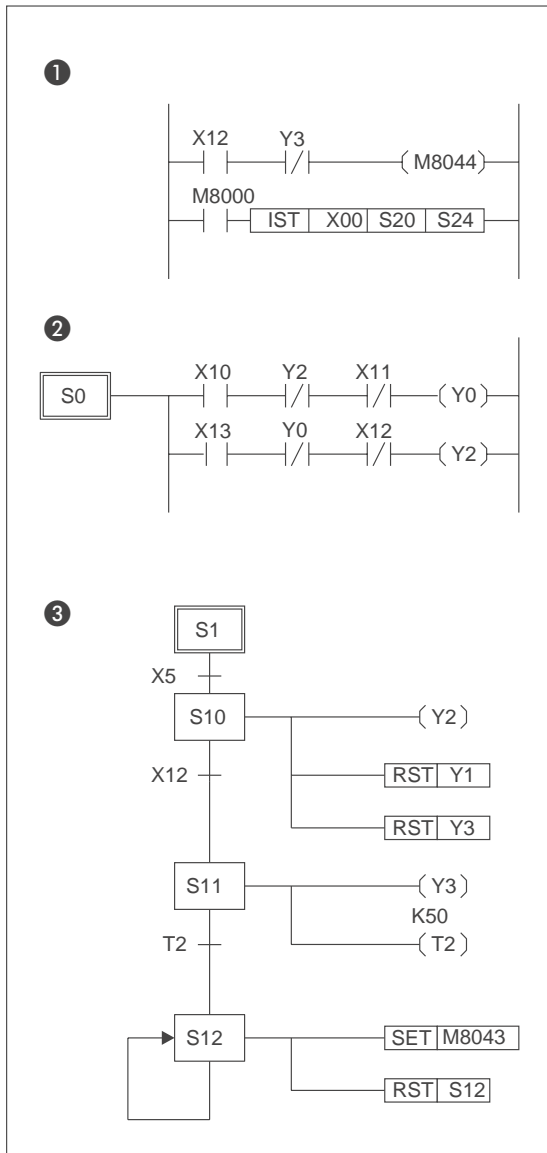


Abb. 6-111:
 Programmierbeispiel zur Be- und Entladekontrolle eines Containerfahrzeuges mit Hilfe der IST-Anweisung.

C000213G

Folgende drei Hauptbereiche sind erforderlich:

- ① Initialisierung
- ② Handbetrieb
- ③ Nullpunktfahrt

Die Schrittkette für den Handbetrieb ermöglicht das manuelle Verfahren des Containerfahrzeuges.

Diese Schrittkette für die Nullpunktfahrt läßt das Containerfahrzeug aus einer beliebigen Position zurück in die Ausgangsposition fahren. Dort wird das Containerfahrzeug entleert.

Beispiel  

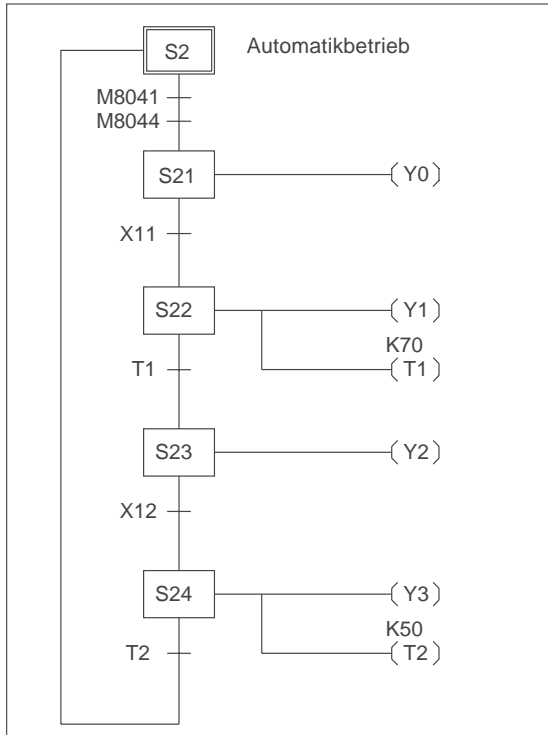


Abb. 6-112:
 Programmierbeispiel zur Be- und Entlade-
 kontrolle eines Containerfahrzeuges mit
 Hilfe der IST-Anweisung (Automatikbetrieb)

C000214G

Initialisierung, Bedingung für erreichten Nullpunkt	Nullposition anfahren	Automatik- betrieb	Hand- betrieb
LD X 12	STL S 1	STL S 2	STL S 0
ANI Y 3	LD X 5	LD M8041	LD X 10
OUT M8044	SET S 10	AND M8044	ANI Y 2
LD M8000	STL S 10	SET S 21	OUT Y 0
IST	RST Y 1	STL S 21	LD X 13
X 0	RST Y 3	OUT Y 0	ANI Y 0
S 20	OUT Y 2	LD X 11	OUT Y 2
S 24	LD X 12	SET S 22	RET
	SET S 11	STL S 22	END
	STL S 11	OUT Y 1	
	OUT Y 3	OUT T 1	
	OUT T 2	K 70	
	K 50	LD T 1	
	LD T 2	SET S 23	
	SET S 12	STL S 23	
	STL S 12	OUT Y 2	
	SET M8043	LD X 12	
	RST S 12	SET S 24	
	RET	STL S 24	
		OUT Y 3	
		OUT T 2	
		K 50	
		LD T 2	
		OUT S 2	
		RET	

C000215G

Abb. 6-113: Programmierbeispiel einer Anweisungsliste zur Be- und Entladekontrolle eines Containerfahrzeuges mit Hilfe der IST-Anweisung



6.8.2 Suchanweisung (SER)

					SER		FNC 61		
					Suchanweisung				CPU
Operanden	S1+	S2+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS, T, C, D	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D	K, H, D		●	16 Bit	32 Bit	SER/SERP
						●	●	DSER/DSERP	17

Funktionsweise

Durchsuchen eines Datenbereichs nach einem Suchwert

Beschreibung

- Der Datenbereich von (S1+) bis ((S1+) + n) wird nach (S2+) durchsucht, und das Suchergebnis wird in den Datenregistern ab (D+) gespeichert. Gleichzeitig wird der kleinste und der größte Wert im Suchbereich ermittelt und gespeichert.
- Die Länge des Datenbereichs n ist für Daten im 16-Bit-Format auf maximal 256 und für Daten im 32-Bit-Format auf maximal 128 festgelegt.
- Das Suchergebnis wird in 5 Datenregistern, bei 32-Bit-Format in 10 Datenregistern, abgelegt. Es enthält:
 - Anzahl der mit dem Suchwert übereinstimmenden Werte im Suchbereich (0 bei keiner Übereinstimmung)
 - die Position des ersten übereinstimmenden Wertes (0 bei keiner Übereinstimmung)
 - die Position des letzten übereinstimmenden Wertes (0 bei keiner Übereinstimmung)
 - die Position des kleinsten in dem Bereich auftretenden Wertes. Tritt dieser Wert mehrfach auf, wird die letzte Position gespeichert.
 - die Position des größten in dem Bereich auftretenden Wertes. Tritt dieser Wert mehrfach auf, wird die letzte Position gespeichert.

Beispiel ▾

Einsatz der SER-Anweisung

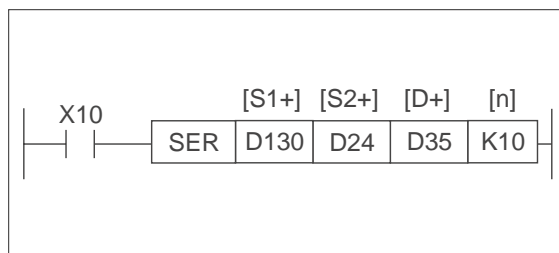


Abb. 6-114:
Programmierbeispiel zum Einsatz der SER-Anweisung

C000326C

Das obige Beispiel zeigt die Suche nach D24 = K100 ab D130 mit Bereichlänge K10. Das Ergebnis wird in D35 bis D39 abgelegt.

Der Suchbereich lässt sich wie folgt darstellen:

Position	Suchliste	Such ergebnis „=“	Maxi- mum	Mini- mum
0	D130 = K100	●		
1	D131 = K111			
2	D132 = K100	●		
3	D133 = K98			
4	D134 = K123			
5	D135 = K66			●
6	D136 = K100	●		
7	D137 = K95			
8	D138 = K78			
9	D139 = K210		●	

Tab. 6-25:
Suchbereich

Das Suchergebnis lässt sich wie folgt darstellen:

Ergebnisliste	Inhalt	Bedeutung
D35	3	Anzahl Suchergebnis „=“
D36	0	Erste Position Übereinstimmung
D37	6	Letzte Position Übereinstimmung
D38	5	Position kleinster Wert
D39	9	Position größter Wert

Tab. 6-26:
Ergebnisliste



6.8.3 Absoluter Counter-Vergleich (ABSD)

					ABSD		FNC 62		
					Absoluter Counter-Vergleich				CPU
						●	●	●	
Operanden	S1+	S2+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS ^① , T, C ^② , D	C ^②	Y, M, S	K, H n ≤ 64		16 Bit	32 Bit	ABSD	9
						●	●	DABSD	17

- ① X, Y, M, S müssen ein Vielfaches von 8 sein.
- ② High-Speed-Counter sind nicht zulässig

Funktionsweise

Schalten von Bits in Abhängigkeit von Zählerständen

Beschreibung

- Der Istwert des Zählers (S2+) wird mit einer Tabelle von Ein- und Ausschaltwerten verglichen.
- Die Tabelle wird aus Wortoperanden gebildet. Der erste Operand ist (S1+). Die Tabelle enthält n Zeilen.

Einschaltwert	Abschaltwert	zu schaltender Operand
(S1+) + 1	(S1+) + 1	D
(S1+) + 2	(S1+) + 3	D + 1
(S1+) + 4	(S1+) + 5	D + 2
⋮	⋮	⋮
(S1+) + (2n+1)	(S1+) + 2n	D + 3

Abb. 6-115:
Tabelle aus Wortoperanden

- Wenn der Zähler (S2+) einen der in der Tabelle hinterlegten Istwerte erreicht, wird der zugeordnete Operand geschaltet.
- Die hinterlegten Werte in der Tabelle dürfen in einem Zahlenbereich von 0 bis 32 767 liegen.
- Die Werte können z. B. mit einer MOV-Anweisung in die Tabelle geschrieben werden.

HINWEISE

Es werden immer 2 Zähler belegt, (S2+) und ((S2+)+1). In (S1+) sollten gerade Operandendressen verwendet werden.

Die ABSD-Anweisung darf nur einmal im Programm genutzt werden.

Beispiel ▾

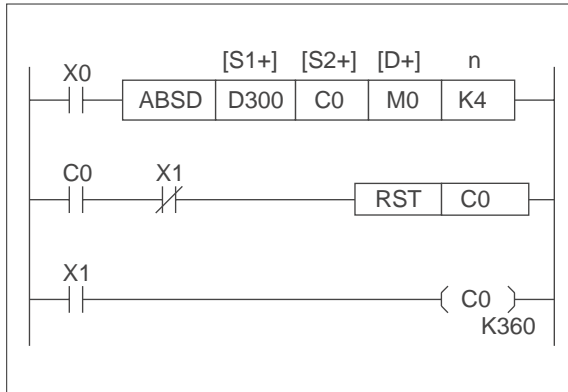


Abb. 6-116:
 Programmierbeispiel zur ABSD-Anweisung

C000163C

Mit dem Programmierbeispiel in der oberen Abb. wird der EIN-/AUS-Status der Merker M0 bis M3 bei einer Drehung eines Rundtisches kontrolliert (siehe auch Abs. 6.8.9).

In (S1+) müssen die Adressen der Operanden X, Y, M und S ein Vielfaches von 8 sein. In (S2+) wird der Counter (C0 bis C199) angegeben. Mit n wird die Anzahl der ein- und auszuschaltenden Zieloperanden (D+) und demzufolge auch die Anzahl der Operanden (S1+), in die die Ein- und Ausschaltwerte geschrieben werden, festgelegt.

Da n = 4, stehen die Merker M0 bis M3 für die Ein- und Ausschaltvorgänge zur Verfügung.

Die vier Einschaltwerte werden in die Datenregister D300, D302, D304 und D306 geschrieben. Die vier Ausschaltwerte werden in die Datenregister D301, D303, D305 und D307 geschrieben.

Für die Einschaltwerte sind die Operanden mit geraden Adressen zu verwenden. Die Ausschaltwerte werden in die Operanden mit ungeraden Adressen geschrieben. Die Ein- und Ausschaltwerte werden mit der MOV-Anweisung in die Datenregister D300 bis D307 geschrieben.

Einschaltwert	Ausschaltwert	Ausgabe
D300 = 40	D301 = 140	M0
D302 = 100	D303 = 200	M1
D304 = 160	D305 = 60	M2
D306 = 240	D307 = 280	M3

Tab. 6-27:
 Ein- und Ausschaltwerte

Wenn X0 eingeschaltet ist, schalten die Merker M0 bis M3 entsprechend der Darstellung in Abb. 6-120. Ist X0 ausgeschaltet, werden die Merker nicht geschaltet.

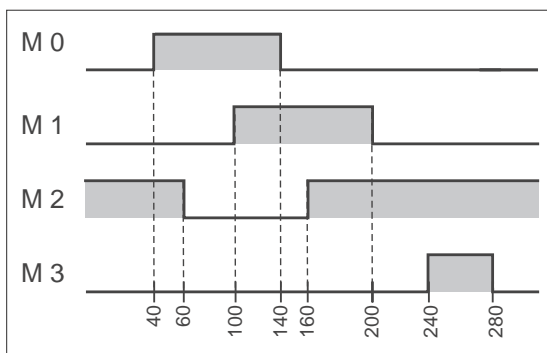


Abb. 6-117:
 Zeitverlauf zum Ein- und Ausschalten der Merker

C000202C



6.8.4 Inkrementaler Counter-Vergleich (INCD)

					INCD		FNC 63		
					Inkrementaler Counter-Vergleich				
					CPU	FX1S	FX1N	FX2N	
						●	●	●	
Operanden	S1+	S2+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	KnX, KnY, KnM, KnS ^① , T, C ^② , D	C ^②	X, Y, M, S	K, H n ≤ 64			16 Bit	32 Bit	INCD
						●			

① X, Y, M, S müssen ein Vielfaches von 8 sein.

② High-Speed-Counter sind nicht zulässig

Funktionsweise

Schalten von Bits in Abhängigkeit von zwei Zählerständen

Beschreibung

- n Bit-Operanden, ausgehend von (D+), werden in Abhängigkeit der Zähler (S2+) und ((S2+)+1) geschaltet.
- In (S1+) werden die Sollwerte für die Einschaltpunkte der Bit-Operanden (D+) vorgegeben.
- Der Zähler (S2+) muß im SPS-Programm programmiert werden; der Sollwert muß größer als der größte Schaltwert in (S1+) sein.
- Der Zähler ((S2+)+1) zählt die Rücksetzvorgänge an (S2+).
- Durch das Ausschalten der Einschaltbedingung der Anweisung werden die Zähler (S2+) und ((S2+)+1) sowie die n Bit-Operanden (D+) zurückgesetzt.
- Nachdem der n. Bit-Operand geschaltet wurde, wird ((S1+)+1) zurückgesetzt und M8029 eingeschaltet.

HINWEIS

Die INCD-Anweisung darf nur einmal im SPS-Programm verwendet werden.

Beispiel ▾

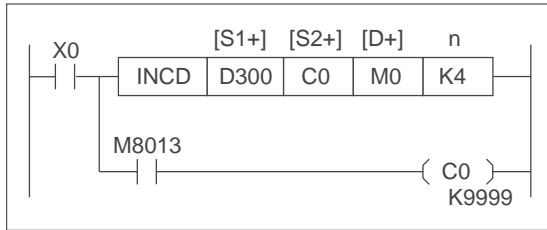


Abb. 6-118:
 Programmierbeispiel zur INCD-Anweisung

C000165C

Setzwert	C0	C1
D300	20	0
D301	30	1
D302	10	2
D303	40	3

Tab. 6-28:
 Beispiel von Setzwerten

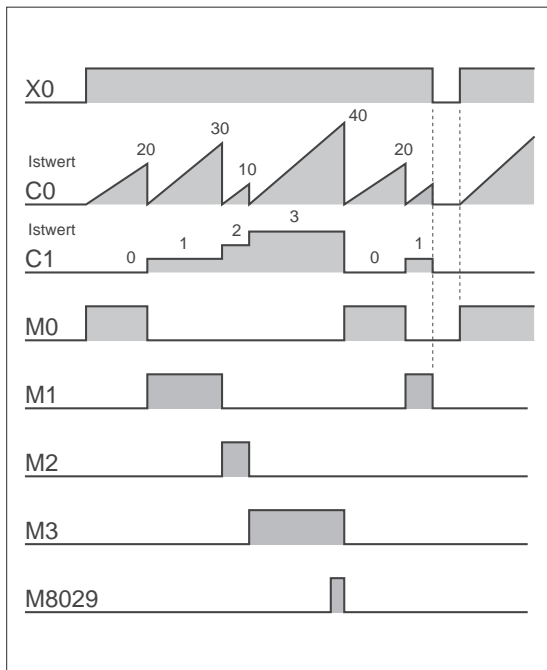


Abb. 6-119:
 Zeitverlauf entsprechend obigen Beispiel zum Ein-/Ausschalten der Merker

C000164C

Der Counter C0 wird automatisch zurückgesetzt, wenn der in D300 bis D303 festgelegte Wert erreicht wurde.

Der Counter C1 zählt die Anzahl der Rücksetzvorgänge an C0.

Die Merker M0 bis M3 schalten entsprechend der Werte des Counters C1.

Das Flag M8029 wird gesetzt, wenn der letzte, d.h. der „n“-te Counter-Prozess, durchgeführt wurde. Anschließend beginnt dieser Vorgang erneut.

Die Counter C0 und C1 werden gelöscht, wenn X0 ausgeschaltet wird; M0 bis M3 werden ebenfalls ausgeschaltet. Wird X0 wieder eingeschaltet, beginnt der Vorgang erneut.

△

6.8.5 Teaching-Timer (TTMR)

			TTMR		FNC 64	
			Teaching-Timer			
Operanden			CPU	FX1S	FX1N	FX2N
						●
	D	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	D	K, H n = 0 – 2		16 Bit	32 Bit	TTMR
			●			

Funktionsweise

Vorgeben eines Timer-Sollwertes über eine Tastenbetätigung

Beschreibung

- Es wird die Einschaltdauer der Anweisung (in Sekunden) gemessen, multipliziert und in das Datenregister ((D+)+1) geschrieben.
- Durch (n) wird der Multiplikator der Zeit festgelegt.

$$n = 0 \rightarrow D+ = [(D+) + 1] \times 1$$

$$n = 1 \rightarrow D+ = [(D+) + 1] \times 10$$

$$n = 2 \rightarrow D+ = [(D+) + 1] \times 100$$

HINWEISE

Durch die Anweisung TTMR werden 2 Datenregister (D+) und ((D+)+1) belegt.

(D+) enthält die multiplizierte Betätigungszeit (s). ((D+)+1) enthält die gemessene Betätigungszeit (s).

Beispiel ▾

Einsatz der TTMR-Anweisung

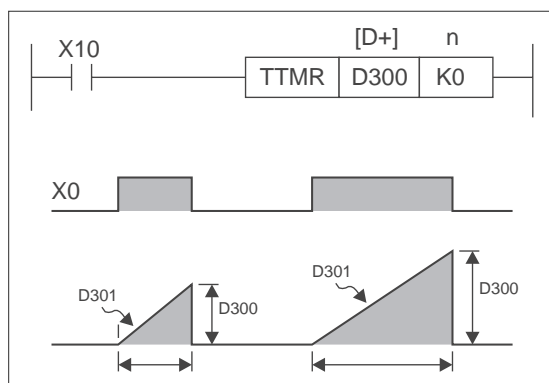


Abb. 6-120:
Programmierbeispiel zur TTMR-Anweisung

Die Betätigungsdauer von X0 wird gemessen.

D300: Betätigungszeit in Sekunden, multipliziert mit 1

D301: Betätigungszeit in Sekunden

C000167C



6.8.6 Sonder-Timer (STMR)

				STMR		FNC 65		
				Sonder-Timer				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T0 – T199	Y, M, S	K, H n=1 – 32767			16 Bit	32 Bit	STMR
					●			

Funktionsweise

Generieren von Sonder-Timer-Funktionen

Beschreibung

- Mit Hilfe der STMR-Anweisung können die Sonder-Timer-Funktionen
 - Ausschaltverzögerung
 - Timer mit Zeitablenkung
 - Blinkgeber

gesteuert werden.

- Der Timer in (S+) steuert die Bit-Operanden (D+) in folgender Art und Weise:

- (D+): Ausschaltverzögerung
- ((D+)+1): Timer mit Zeitablenkung
- ((D+)+2) / ((D+)+3): Blinkgeber

Es werden, ausgehend von (D+), vier aufeinanderfolgende Bits belegt.

HINWEIS

Die in dieser Anweisung eingesetzten Timer dürfen nicht an anderer Stelle nochmals verwendet werden.

Beispiel ▾

Einsatz der STMR-Anweisung

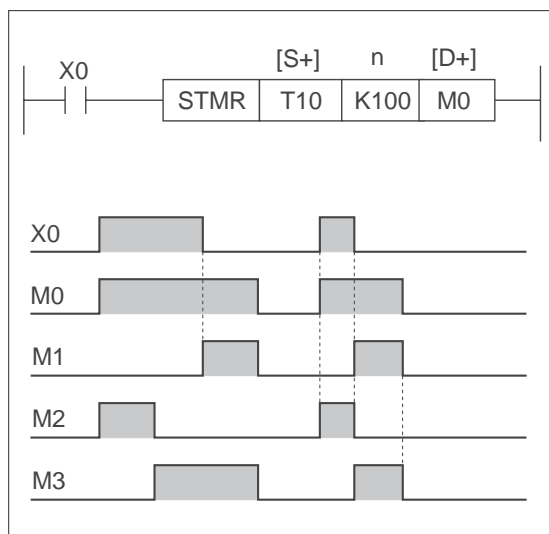


Abb. 6-121:
 Programmierbeispiel zum Generieren von Sonder-Timer-Funktionen

C000161C



6.8.7 Flip-Flop-Funktion (ALT)

		ALT		FNC 66		
		Flip-Flop-Funktion				
Operanden	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	Y, M, S		●	16 Bit		32 Bit
				●		ALT/ALTP

Funktionsweise

Realisierung der Flip-Flop-Funktion

Beschreibung

- Mit der ALT-Anweisung können Sie eine Flip-Flop-Funktion programmieren.
- Die ALT-Anweisung wird durch Ansteuern mit einem „1“-Signal aktiviert, und der in D+ angegebene Operand wird gesetzt.
- Durch erneutes Ansteuern mit einem „1“-Signal wird der Operand zurückgesetzt.

HINWEIS

Die Anweisung wird in jedem Programmzyklus ausgeführt. Dies können Sie durch Einsatz einer vorgeschalteten Impulsfunktion (PLS-Anweisung) oder des Parameters „P“ verhindern.

Beispiel ▾

Einsatz der ALT-Anweisung, Operandenstatus invertieren

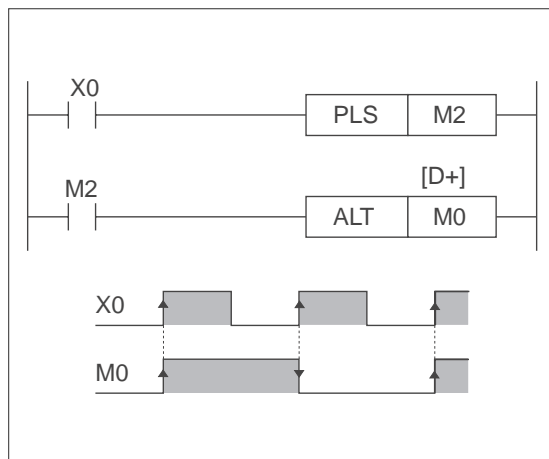


Abb. 6-122: Programmierbeispiel zum Einsatz der ALT-Anweisung (Operandenstatus invertieren)

C000108C

Der Status des Merkers M0 wird jedesmal invertiert, wenn am Eingang X0 ein „1“-Signal ansteht.



Beispiel ▾ Einsatz der ALTP-Anweisung, Operandenstatus invertieren

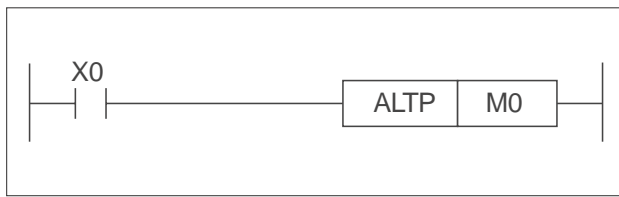


Abb. 6-123:
 Programmierbeispiel zum Einsatz der ALTP-Anweisung (Operandenstatus invertieren)

Die Funktion ist identisch mit der in Abb. 6-122.



Beispiel ▾ Einsatz der ALT-Anweisung, Start-Stopp-Funktion

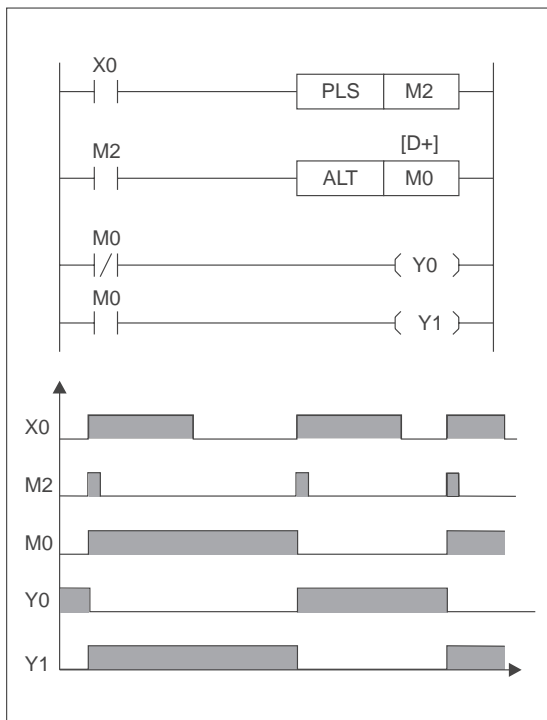


Abb. 6-124:
 Programmbeispiel zum Einsatz der ALT-Anweisung (Start-Stopp-Funktion)

Der Start-Ausgang Y1 wird durch Betätigen des Tasters X0 aktiviert. Der Stopp-Ausgang Y0 wird durch nochmaliges Betätigen des Tasters X0 aktiviert.

C000114C



6.8.8 Rampenfunktion (RAMP)

			RAMP		FNC 67		
			Rampenfunktion				
			CPU	FX1S	FX1N	FX2N	
				●	●	●	
Operanden	S1+, S2+, D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D	K, H n=1 bis +32 767		16 Bit	32 Bit	RAMP	9
			●				

Funktionsweise

Zeitabhängige Veränderung eines Datenwertes, ausgehend von einem Start- zu einem Zielwert

Beschreibung

- In dem in (S1+) angegebenen Datenregister wird ein Ausgangswert festgelegt.
- In dem in (S2+) angegebenen Datenregister wird ein Zielwert festgelegt.
- In dem in (D+) angegebenen Datenregister wird ein aktueller Rampenfunktionswert abgespeichert. Als Startwert wird in (D+) der Ausgangswert gespeichert. Der Rampenfunktionswert wird n Mal verändert, bis der Zielwert erreicht ist.

(S1+): Ausgangswert
 (S2+): Zielwert
 (D+): aktueller Rampenfunktionswert
 n: Anzahl der Operationszyklen

- Die benötigte Ausführungszeit T beträgt: $T = (n \times \text{Programmzykluszeit})$
- Ist in (D+) der Zielwert erreicht, wird der Sondermerker (Flag) M8029 gesetzt. Der Zielwert bleibt weiterhin in (D+) gespeichert.
- Die Anzahl der Operationszyklen wird nach Abarbeitung der Anweisung in dem auf (D+) folgenden Datenregister gespeichert.

HINWEISE

Die Ausführungszeit der RAMP-Anweisung ist von der Programmzykluszeit abhängig. Deshalb sollte die Steuerung mit einer konstanten Programmzykluszeit betrieben werden, um ein definiertes Verhalten der Funktion zu gewährleisten.

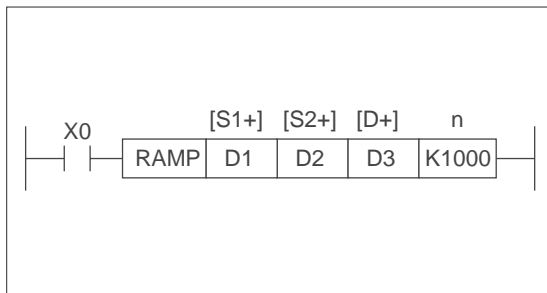
Bei FX/FX2N-CPU's kann der Verlauf der RAMP-Funktion mit Hilfe des Sondermerkers M8026 beeinflusst werden. Ist M8026 nicht gesetzt erfolgt eine ständige Wiederholung der Rampenfunktion. Das bedeutet, wenn der Istwert von D gleich dem Wert in S2 ist, wird die Rampenfunktion automatisch zurückgesetzt und erneut gestartet. Bei gesetztem Sondermerker M8026 wird die Rampenfunktion gehalten. Das heißt, sobald der Istwert von D dem Wert von S2 entspricht behält die Rampenfunktion den augenblicklichen Zustand bei. M8029 bleibt in diesem Fall solange gesetzt wie die Rampenfunktion aktiv ist. Der Wert in D wird solange nicht zurückgesetzt bis die Anweisung re-initialisiert ist.

Bei FX1S- und FX1N-CPU's kann der Verlauf der RAMP-Funktion nicht beeinflusst werden. Hier verhält sich die Rampenfunktion so als wenn M8026 gesetzt wäre, d.h. die Rampenfunktion behält den augenblicklichen Zustand bei.

Wird die Rampenfunktion vor Beendigung unterbrochen, wird die Position zum Zeitpunkt der Unterbrechung solange beibehalten, bis das Auslösesignal wieder ansteht. Liegt das Rampensignal wieder an, werden die Register D + D1 zurückgesetzt und der Zyklus beginnt von neuem.

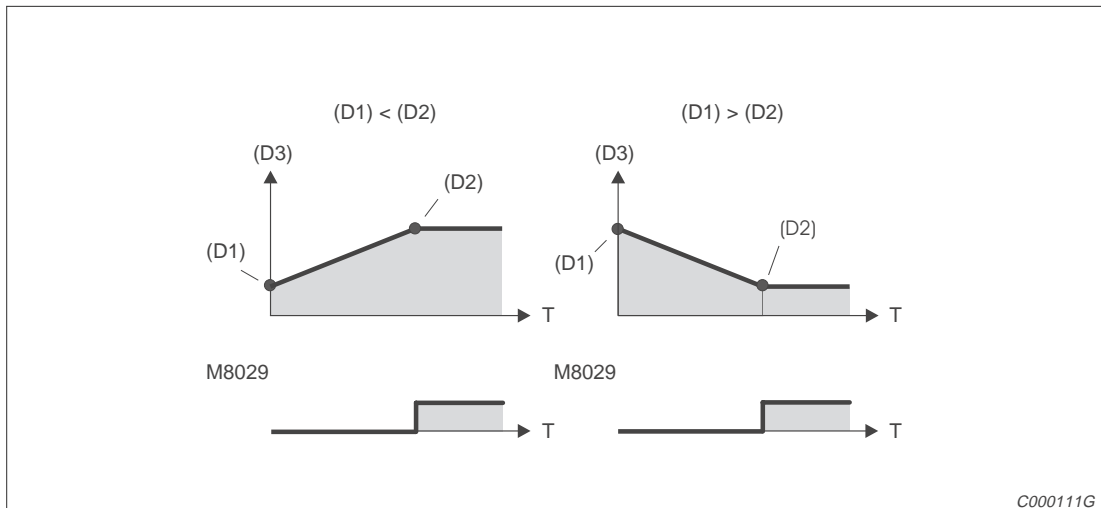
Beispiel ▾

Einsatz der RAMP-Anweisung

**Abb. 6-125:**

Programmierbeispiel zum Einsatz der RAMP-Anweisung

C000099C



C000111G

Abb. 6-126: Zeitverläufe zum obigen Beispiel

- Nach Einschalten von X0 nimmt D3 zunächst den in D1 festgelegten Ausgangswert an. Dieser Wert wird fortlaufend (1000 Mal) verändert, bis der in D2 festgelegte Zielwert erreicht ist.
- Die für diesen Vorgang benötigte Zeit T beträgt: $T = (n \times \text{Programmzykluszeit})$.
- Die Anzahl der Operationszyklen n wird in D4 abgespeichert.
- Wird nach dem Festlegen der Programmzykluszeit (die etwas länger ist als die aktuelle Programmzykluszeit) im Datenregister D8039 der Sondermerker M8039 aktiviert, arbeitet die SPS mit einer konstanten Programmzykluszeit.

Beträgt z. B. der in D8039 festgelegte Wert 20 ms, beansprucht die Änderung im Datenregister D3 vom Ausgangswert bis zum Erreichen des Zielwertes $T = 1000 \times 20 \text{ ms} = 20 \text{ s}$.

- Ist X0 ausgeschaltet, wird die Ausführung der Rampenfunktion abgebrochen. Wird danach X0 wieder eingeschaltet, beginnt die Ausführung der Rampenfunktion erneut mit dem Ausgangswert.
- Ist die Ausführung der Rampenfunktion beendet, wird der Sondermerker (Flag) M8029 gesetzt, und D3 nimmt den in D1 festgelegten Ausgangswert an.
- Stellen Sie sicher, daß D4 gelöscht wird, wenn die SPS nach einem Stillstand wieder in den RUN-Modus geschaltet wird und X0 noch gesetzt ist.

△

HINWEIS

Die Rampenfunktion kann auch mit den Sondermerkern M8193 und M8194 verwendet werden um die Funktion der SER- und RS-Anweisung (FNC61 und FNC 80) zu simulieren. Dies ist in erster Linie bei älteren Programmiergeräten von Bedeutung.

6.8.9 Rundtisch-Positionierung (ROTC)

				ROTC		FNC 68		
				Rundtisch-Positionierung				
				CPU	FX1S	FX1N	FX2N	
Operanden	S+	m1 / m2	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D ①	K, H ②	Y, M, S ③		●	16 Bit	32 Bit	ROTC

① 3 aufeinanderfolgende Datenregister ($((S+1) \leq m1)$)

② $m1 = 2$ bis 32 767; $m2 = 0$ bis 32 767 ($m1 \geq m2$)

③ 8 aufeinanderfolgende Bits

Funktionsweise

Steuern eines Rundtisches

Beschreibung

- Die Position des Rundtisches wird über einen 2-Phasen-Encoder erfasst.
- Alle Zielpositionen auf dem Tisch werden relativ zu einer Nullposition angegeben.
- Die Zielposition wird immer auf dem kürzesten Weg angefahren.

HINWEIS

Folgende Operanden werden durch die Anweisung geschaltet bzw. abgefragt:

((D+)+0)	A-Phasensignal des Zählers
((D+)+1)	B-Phasensignal des Zählers
((D+)+2)	Nullphasensignal des Zählers
((D+)+3)	Ausgabe: schnelle Tischdrehung vorwärts
((D+)+4)	Ausgabe: Schleichfahrt vorwärts
((D+)+5)	Stopp-Ausgang
((D+)+6)	Ausgabe: Schleichfahrt rückwärts
((D+)+7)	Ausgabe: schnelle Tischdrehung rückwärts
m1	Anzahl Zählimpulse pro Tischumdrehung
m2	Anzahl Zählimpulse für die in Schleichfahrt zurückzulegende Strecke
((S+)+0)	Istposition (kann nur gelesen werden)
((S+)+1)	Zielposition
((S+)+2)	Teilekennung des zu verfahrenen Teils

Beispiel ▾

Erfassung der Zählimpulse

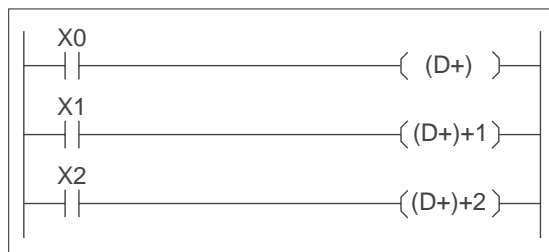


Abb. 6-127:
Programmierbeispiel für Erkennungssignale

C000238C

Da der Encoder und der Schalter für das Nullsignal über Eingänge erfasst werden müssen, ist es notwendig, diese Eingänge auf die Operanden (D+) zu schreiben. △

HINWEISE

- | Die Parameter ((S+)+1) und ((S+)+2) müssen vor dem Einschalten beschrieben werden.
- | Vor der ersten Inbetriebnahme sollte der Tisch in seine Nullposition gebracht werden.
- | Die Anweisung kann nur einmal im Programm eingesetzt werden.
- | Da die Drehbewegung des Tisches über normale Eingänge erfasst wird, dürfen nicht mehr als ca. 25 Impulse/s ausgegeben werden.

Beispiel ▾

Ein Rundtisch benötigt für eine Umdrehung 500 Impulse. Der Tisch verfügt über 10 Stationen; daraus folgt, daß 50 Impulse zwischen zwei Stationen liegen. Die Station Null wird als „Nullposition“ aufgefasst.

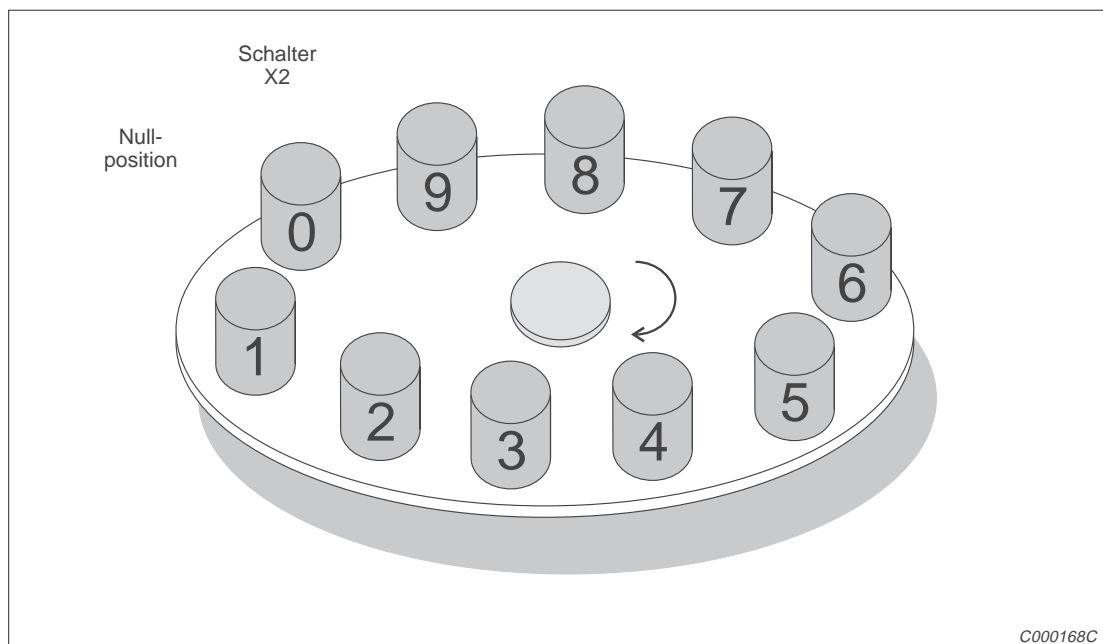


Abb. 6-128: Beispiel für eine Rundtischpositionierung

Wenn das Teil, das an Position 7 auf dem Tisch liegt, an die Position 3 gefahren werden soll, muß die ROTC-Anweisung wie folgt aussehen:

(S+) = beliebiges Datenregister, z.B. D200
 ((S+)+1) = $3 \times 50 = 150$: Entfernung der Position in Impulsen von der Station 0.
 Wert muß vor der Ausführung der Anweisung in D201 gespeichert werden.
 ((S+)+2) = $7 \times 50 = 350$: Entfernung der Position in Impulsen von der Station 0.
 Wert muß vor der Ausführung der Anweisung in D201 gespeichert werden.
 m1 = 500

Wenn mit zwei Geschwindigkeiten gefahren werden soll, muß der Weg, der langsam zurückgelegt werden soll, in (m2) Impulsen angegeben werden.

m2 = 0 (keine Schleichfahrt)
 (D+) = beliebige Bits M, Y, S, über die der Motor des Tisches angesteuert wird.

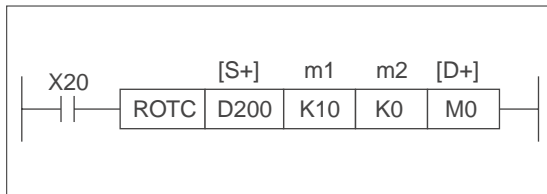

Beispiel 

Abb. 6-129:
Programmierbeispiel für eine ROTC-
Anweisung

C000240C

Wenn X20 angeschaltet wird, wird der Tisch um 4 Positionen nach rechts verdreht.

Der Motor wird über den Merker M3 gestartet. Wenn die Position erreicht ist, wird der Merker M5 eingeschaltet.

Über die Merker M0, M1 und M2 wird die Drehung des Tisches überwacht. 

6.8.10 Sortieranweisung (SORT)

					SORT		FNC 69		
					Sortieranweisung				CPU
Operanden	S+	n1 / n2	D+	m	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D ①	K, H ②	D	K, H, D		16 Bit	32 Bit	SORT	11

- ① ((n1) x (n2)) aufeinanderfolgende Datenregister
- ② n1 = 1 bis 32; n2 = 1 bis 6

Funktionsweise

Sortieren einer Matrix nach den Werten einer Spalte

Beschreibung

Bei Aufruf der SORT-Anweisung wird eine interne Datenmatrix, gekennzeichnet durch das Start-Datenregister (S+) mit der Größe von n1 Zeilen und n2 Spalten, nach den Werten in Spalte m sortiert und ab Datenregister (D+) neu abgelegt.

HINWEISE

- | Die SORT-Anweisung darf nur **einmal** im Programm verwendet werden.
- | Bei Ausführung der SORT-Anweisung wird jeder Eintrag entsprechend den Daten im ausgewählten Sortierfeld m in aufsteigender Folge sortiert.
- | (S+) und (D+) können das gleiche Datenregister angeben, da die gespeicherten Werte nicht verändert werden.
- | Überschneiden sich bei unterschiedlichen Datenregistern (S+) und (D+) die Datenbereiche, in denen die Matrix gespeichert ist, kann es zu einem Datenverlust kommen.
- | Nach Ausführung einer SORT-Anweisung wird mit M8029 eine Kennung gesetzt. Ein Sortiervorgang ist erst dann abgeschlossen, wenn die in n1 vorgegebene Anzahl erreicht ist.
- | Während eines Sortiervorgangs dürfen die Daten in der Sortiermatrix nicht verändert werden, da andernfalls fehlerhafte Daten abgelegt werden.

Beispiel ▾ Anwendung der SORT-Anweisung auf eine Matrix

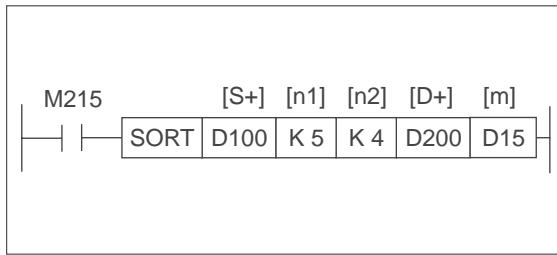


Abb. 6-130:
 Programmierbeispiel zum Einsatz der SORT-Anweisung

C000327C

Die Datenmatrix könnte die folgende Form aufweisen:

Spalte Nr.	1	2	3	4
Zeile Nr.	ID-Nummer	Höhe	Gewicht	Alter
1	D100	D105	D110	D115
	1	150	45	20
2	D101	D106	D111	D116
	2	180	50	40
3	D102	D107	D112	D117
	3	160	70	30
4	D103	D108	D113	D118
	4	100	20	8
5	D104	D109	D114	D119
	5	150	50	45

Tab. 6-29:
 Unsortierte Matrix

Sortiert nach Spalte (m) = K2 ergibt sich:

Spalte Nr.	1	2	3	4
Zeile Nr.	ID-Nummer	Höhe	Gewicht	Alter
1	D200	D205	D210	D215
	4	100	20	8
2	D201	D206	D211	D216
	1	150	45	20
3	D202	D207	D212	D217
	5	150	50	45
4	D203	D208	D213	D218
	3	160	70	30
5	D204	D209	D214	D219
	2	180	50	40

Tab. 6-30:
 Sortierte Matrix nach Anwendung der SORT-Anweisung

7 Spezielle FNC-Anweisungen

7.1 Allgemeine Hinweise

Dieses Kapitel beschreibt die speziellen FNC-Anweisungen der FX-Familie für besondere Anwendungen. Mit diesen Anweisungen lassen sich Funktionen zur Datenein- und -ausgabe, zur Modul-Kommunikation und zur Steuerung von Sondermodulen realisieren.

Eine einführende Erläuterung zur Struktur der Anweisungstabellen enthält Abschnitt 6.1.1.

7.1.1 Gesamtübersicht der speziellen FNC-Anweisungen

Einteilung	Anweisung	FNC	Bedeutung	Referenz	Steuerung		
					FX1S	FX1N	FX2N
Ein-/Ausgabe-anweisungen	TKY	70	Zehnertastatur	7.2.1			●
	HKY	71	Hexadezimale Tastatur	7.2.2			●
	DSW	72	Digitaler Schalter	7.2.3	●	●	●
	SEGD	73	7-Segment-Anzeige	7.2.4			●
	SEGL	74	7-Segment-Anzeige mit Latch	7.2.5	●	●	●
	ARWS	75	7-Segment-Anzeige mit zusätzlichen Tasten	7.2.6			●
	ASC	76	ASCII-Konvertierung	7.2.7			●
	PR	77	Datenausgabe über die Ausgänge	7.2.8			●
	FROM	78	Auslesen von Daten aus einem Sondermodul	7.2.9		●	●
	TO	79	Schreiben von Daten in ein Sondermodul	7.2.10		●	●
Anweisungen für serielle Kommunikation	RS	80	Serielle Datenübertragung	7.3.1	●	●	●
	PRUN	81	Umliegen von Eingängen oder Merkern	7.3.2	●	●	●
	ASCI	82	Umwandlung in ein ASCII-Zeichen	7.3.3	●	●	●
	HEX	83	Umwandlung in einen Hexadezimalwert	7.3.4	●	●	●
	CCD	84	Summen- und Paritätsprüfung	7.3.5	●	●	●
	VRRD	85	Einlesen von Sollwerten vom FX1N-8AV-BD und FX2N-8AV-BD	7.3.6	●	●	●
	VRSC	86	Einlesen von Schalterstellungen vom FX1N-8AV-BD und FX2N-8AV-BD	7.3.7	●	●	●
	PID	88	Programmierung eines Regelkreises	7.3.8	●	●	●

Tab. 7-1: Übersicht der speziellen FNC-Anweisungen

Einteilung	Anweisung	FNC	Bedeutung	Referenz	Steuerung		
					FX1S	FX1N	FX2N
Anweisungen mit Gleitkommazahlen	DECMP	110	Vergleich von Gleitkommazahlen	7.4.1			●
	DEZCP	111	Vergleich von Gleitkommazahlen mit einem Bereich	7.4.2			●
	DEBCD	118	Umwandlung des Gleitkommaformats ins wissenschaftliche Zahlenformat	7.4.3			●
	DEBIN	119	Umwandlung des wissenschaftlichen Zahlenformats ins Gleitkommaformat	7.4.4			●
	DEADD	120	Addition von Gleitkommazahlen	7.4.5			●
	DESUB	121	Subtraktion von Gleitkommazahlen	7.4.6			●
	DEMUL	122	Multiplikation von Gleitkommazahlen	7.4.7			●
	DEDIV	123	Division von Gleitkommazahlen	7.4.8			●
	DESQR	127	Quadratwurzeln aus Gleitkommazahlen	7.4.9			●
	INT	129	Umwandlung des Gleitkommaformats ins Dezimal-Format	7.4.10			●
	DSIN	130	Sinusberechnung mit Gleitkommazahlen	7.4.11			●
	DCOS	131	Cosinusberechnung mit Gleitkommazahlen	7.4.12			●
	DTAN	132	Tangensberechnung mit Gleitkommazahlen	7.4.13			●
Datenverarbeitungsanweisungen	SWAP	147	High-Low-Byte-Tausch	7.5.1			●
Positionierung mit Servoverstärker	ABS	155	Lesen der absoluten Ist-Position	7.6.1	●	●	
	ZRN	156	Referenzpunkt anfahren	7.6.2	●	●	
	PLSV	157	Ausgabe von Impulsen mit variabler Frequenz	7.6.3	●	●	
	DRVI	158	Positionieren auf einen Inkrementalwert	7.6.4	●	●	
	DRVA	159	Positionieren auf einen Absolutwert	7.6.5	●	●	
Echtzeituhranweisungen	TCMP	160	Vergleich von Uhr-Daten	7.7.1	●	●	●
	TZCP	161	Vergleich von Uhr-Daten mit einem Bereich	7.7.2	●	●	●
	TADD	162	Addition von Uhr-Daten	7.7.3	●	●	●
	TSUB	163	Subtraktion von Uhr-Daten	7.7.4	●	●	●
	TRD	166	Lesen von Uhr-Daten	7.7.5	●	●	●
	TWR	167	Schreiben von Uhr-Daten	7.7.6	●	●	●
	HOUR	169	Betriebsstundenzähler	7.7.7	●	●	
Gray-Code-Anweisungen	GRY	170	Umwandlung Integer in Gray-Code	7.8.1			●
	GBIN	171	Umwandlung Gray-Code in Integer	7.8.2			●
Datenaustausch mit FX0N-3A	RD3A	176	Analogwert von FX0N-3A lesen	7.9.1		●	
	WR3A	177	Daten zum FX0N-3A übertragen	7.9.2		●	

Tab. 7-2: Übersicht der speziellen FNC-Anweisungen

Einteilung	Anweisung	FNC	Bedeutung	Referenz	Steuerung		
					FX1S	FX1N	FX2N
LADE-verknüpfte Vergleiche	LD=	224	Vergleichsanweisung, gleich	7.10.1	●	●	●
	LD>	225	Vergleichsanweisung, größer	7.10.1	●	●	●
	LD<	226	Vergleichsanweisung, kleiner	7.10.1	●	●	●
	LD<>	228	Vergleichsanweisung, ungleich	7.10.1	●	●	●
	LD≤	229	Vergleichsanweisung, kleiner gleich	7.10.1	●	●	●
	LD≥	230	Vergleichsanweisung, größer gleich	7.10.1	●	●	●
UND-verknüpfte Vergleiche	AND=	232	UND-verknüpfte Vergleichsanweisung, gleich	7.10.2	●	●	●
	AND>	233	UND-verknüpfte Vergleichsanweisung, größer	7.10.2	●	●	●
	AND<	234	UND-verknüpfte Vergleichsanweisung, kleiner	7.10.2	●	●	●
	AND<>	236	UND-verknüpfte Vergleichsanweisung, ungleich	7.10.2	●	●	●
	AND≤	237	UND-verknüpfte Vergleichsanweisung, kleiner gleich	7.10.2	●	●	●
	AND≥	238	UND-verknüpfte Vergleichsanweisung, größer gleich	7.10.2	●	●	●
ODER-verknüpfte Vergleiche	OR=	240	ODER-verknüpfte Vergleichsanweisung, gleich	7.10.3	●	●	●
	OR>	241	ODER-verknüpfte Vergleichsanweisung, größer	7.10.3	●	●	●
	OR<	242	ODER-verknüpfte Vergleichsanweisung, kleiner	7.10.3	●	●	●
	OR<>	244	ODER-verknüpfte Vergleichsanweisung, ungleich	7.10.3	●	●	●
	OR≤	245	ODER-verknüpfte Vergleichsanweisung, kleiner gleich	7.10.3	●	●	●
	OR≥	246	ODER-verknüpfte Vergleichsanweisung, größer gleich	7.10.3	●	●	●

Tab. 7-3: Übersicht der speziellen FNC-Anweisungen

7.2 Ein-/Ausgabeeweisungen

Übersicht der Anweisungen FNC 70 bis 79

Symbol	FNC	Bedeutung	Abschnitt
TKY	70	Zehnerastatur	7.2.1
HKY	71	Hexadezimale Tastatur	7.2.2
DSW	72	Digitaler Schalter	7.2.3
SEGD	73	7-Segment-Anzeige	7.2.4
SEGL	74	7-Segment-Anzeige mit Latch	7.2.5
ARWS	75	7-Segment-Anzeige mit zusätzlichen Tasten	7.2.6
ASC	76	ASCII-Konvertierung	7.2.7
PR	77	Datenausgabe über die Ausgänge	7.2.8
FROM	78	Auslesen von Daten aus einem Sondermodul	7.2.9
TO	79	Schreiben von Daten in ein Sondermodul	7.2.10

Tab. 7-4: Übersicht der Anweisungen FNC 70 bis 79

7.2.1 Zehnertastatur (TKY)

				TKY		FNC 70		
				Zehnertastatur				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S1+	D1+	D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S ①	KnY, KnM, KnS, T, C, D, V, Z	Y, M, S ②		16 Bit	32 Bit	TKY	7
					●	●	DTKY	13

- ① 10 aufeinanderfolgende Bits
- ② 11 aufeinanderfolgende Bits

Funktionsweise

Einlesen einer Zehnertastatur über die Eingänge

Beschreibung

- Eine Tastatur mit 10 Tasten wird über die Bits (S+) bis ((S+)+9) in die Steuerung eingelesen.
- Die eingegebenen Werte werden nacheinander im Datenwort (D1+) abgelegt. Bei einer 16-Bit-Operation können 4 Stellen (max. 9.999) und bei einer 32-Bit-Operation 8 Stellen (max. 99.999.999) geschrieben werden.
- Wenn mehr als die möglichen 4 bzw. 8 Stellen eingegeben werden, werden nur die zuletzt eingegebenen 4 bzw. 8 Stellen in (D1+) gespeichert.
- Die Bits (D2+) bis ((D2+)+10) spiegeln den Status der Tasten wieder.

HINWEISE

Die TKY-Anweisung darf nur einmal im SPS-Programm eingesetzt werden.

Wenn die Anweisung TKY nicht mehr aktiv ist, werden die Bits (D2+) gelöscht. Der Inhalt von (D1+) bleibt erhalten.

Beispiel ▾

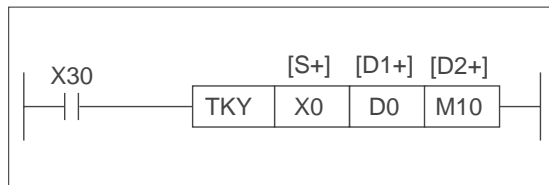


Abb. 7-1:
Programmierbeispiel zur TKY-Anweisung

C000242C

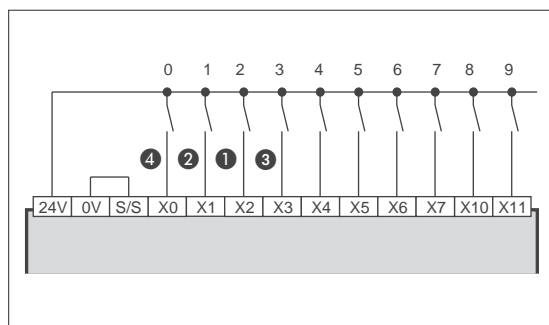


Abb. 7-2:
Zuordnung der Tasten

C000241C

Im Beispiel werden den numerischen Tasten 0 bis 9 Eingänge X zugeordnet. In (S+) wird die Startadresse X0 angegeben.

Beispiel 

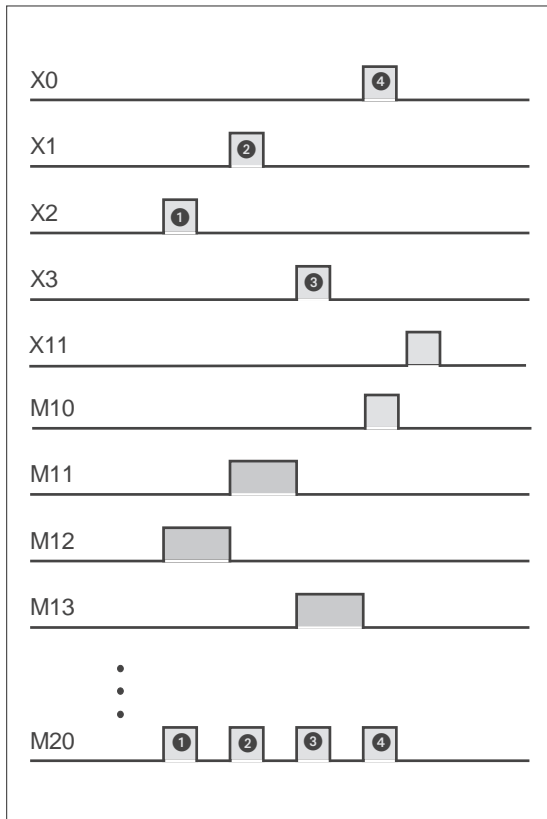


Abb. 7-3:
 Programmierbeispiel zum Ein-/Ausschalten der Eingänge und Merker

C000243C

Wenn die Tasten X0 bis X3 in der Reihenfolge ① bis ④ betätigt werden, steht in dem Datenregister D0 der Wert 2 130. Wenn danach die Taste X11 betätigt wird, fällt die 2 aus dem Register heraus, und der neue Inhalt von D0 beträgt 1 309. 

7.2.2 Hexadezimale Tastatur (HKY)

					HKY		FNC 71		
					Hexadezimale Tastatur				CPU
Operanden	S+	D1+	D2+	D3+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X	Y	T, C, D, V, Z	Y, M, S ^①		16 Bit	32 Bit	HKY	9
						●	●	DHKY	17

① 8 aufeinanderfolgende Bits

Funktionsweise

Einlesen einer hexadezimalen Tastatur über die Eingänge

Beschreibung

- Die Tasten werden in einem Multiplex-Verfahren eingelesen. Es werden jeweils 4 Ein- und Ausgänge belegt.
- Die Tasten 0 bis 9 werden als Zahl aufgefaßt und in (D2+) eingetragen.
- Die Tasten A bis F schalten die Bits (D3+) bis ((D3+)+5).
- (D3+) gibt den ersten von 8 Merkern zur Speicherung der Funktionstastenbetätigung und der Kontrollsignale an. Die Tasten A bis F schalten die Merker (D3+) bis ((D3+)+5). Merker ((D3+)+6) wird bei Betätigung einer der Tasten A bis F belegt und Merker ((D3+)+7) bei Betätigung einer der Tasten 0 bis 9. Nach jeder Registrierung einer Tastenbetätigung wird Merker M8029 gesetzt.
- Die durch die Tasten 0 bis 9 eingegebene Zahl wird in (D2+) abgelegt. Es können maximal 4 Stellen eingegeben werden (max. 9.999).
Wenn die 32-Bit-Operation ausgeführt wird, können 8 Stellen (max. 99.999.999) eingegeben werden.
- Werden mehr als eine Taste betätigt, wird die zuerst gedrückte Taste berücksichtigt.
- Wenn mehr als 4 bzw. 8 Stellen eingegeben werden, werden nur die zuletzt eingegebenen 4 bzw. 8 Stellen berücksichtigt.

HINWEISE

Die HKY-Anweisung darf nur einmal im Programm eingesetzt werden.

Bei der Verwendung der HKY-Anweisung sollte die Steuerung mit konstanter Zykluszeit betrieben werden.

Beispiel ▾

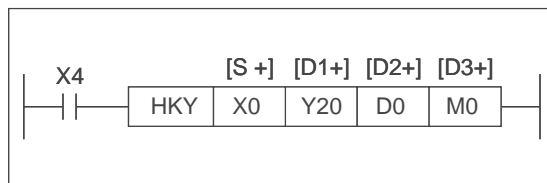


Abb. 7-8:
Programmierbeispiel zur HKY-Anweisung

C000244C

Beispiel 

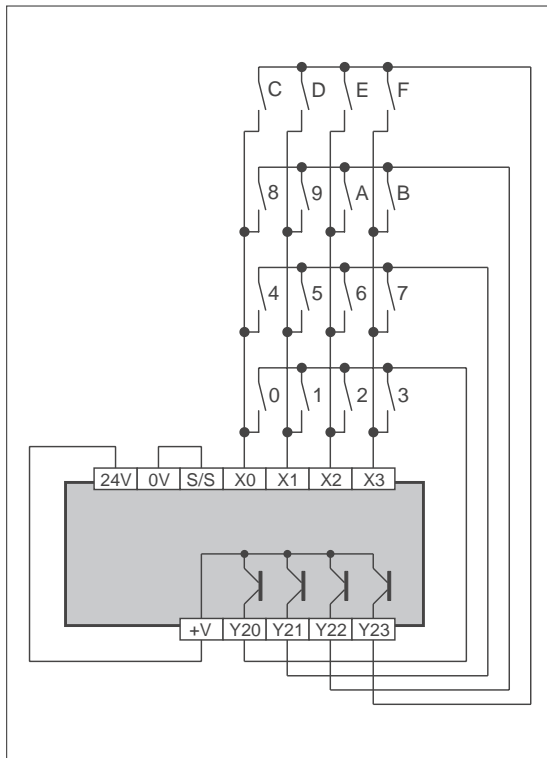


Abb. 7-6:
 Programmierbeispiel zum Ein-/Ausschalten
 der Eingänge und Merker

C000245C

Funktionstasten

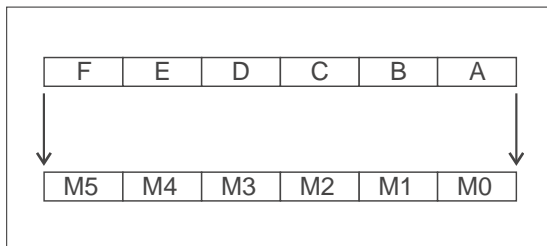


Abb. 7-7:
 Beispiel einer Merker-Zuordnung

C000246C

Wird die Taste A betätigt, wird der Merker M0 gesetzt. M0 bleibt solange eingeschaltet, bis eine andere Taste betätigt wird. Wird anschließend die Taste D betätigt, wird M0 ausgeschaltet und M3 eingeschaltet.

Werden zwei oder mehr Tasten betätigt, wird die zuerst gedrückte Taste berücksichtigt.

Ausgänge

Solange eine der Tasten A bis F gedrückt gehalten wird, wird M6 aktiviert.

Solange eine der Tasten 1 bis 9 gedrückt gehalten wird, wird M7 aktiviert.

Ist X4 ausgeschaltet, werden die Daten in D0 nicht verändert. Die Merker M0 bis M7 werden ausgeschaltet.

Das Erfassen der Tastenbetätigung erfordert 8 Zyklen.

Datenspeicher

Der eingegebene Wert wird vierstellig im Datenregister D0 abgelegt.



Funktionsweise mit Sondermerker M8167

Einlesen einer hexadezimalen Tastatur über die Eingänge

Beschreibung

- Das Setzen des Sondermerkers M8167 bewirkt, daß die Eingabe über die 16 Tasten (1 – 9, A – F) als Hexadezimal-Format interpretiert wird.
- Die Tasten werden in einem Multiplex-Verfahren eingelesen. Es werden jeweils 4 Ein- und Ausgänge belegt, wobei (S+) den ersten Eingang und (D1+) den ersten Ausgang angibt.
- Die Tasten werden als Zahl aufgefaßt und in (D2+) eingetragen.
- Die durch die Tasten 0 bis 9 und A bis F eingegebene Zahl wird in (D2+) abgelegt. Es können maximal 4 Stellen eingegeben werden (max. FFFF_H).
Wenn die 32-Bit-Operation ausgeführt wird, können 8 Stellen (max. FFFFFFFF_H) eingegeben werden.
- Werden mehr als eine Taste betätigt, wird die zuerst gedrückte Taste berücksichtigt.
- Wenn mehr als 4 bzw. 8 Stellen eingegeben werden, werden nur die zuletzt eingegebenen 4 bzw. 8 Stellen berücksichtigt.

HINWEISE

| Die HKY-Anweisung darf nur einmal im Programm eingesetzt werden.

| Bei der Verwendung der HKY-Anweisung sollte die Steuerung mit einer konstanten Zykluszeit von mehr als 20 ms betrieben werden.

Ist die Zykluszeit zu kurz, sollten Sie mit einem Timer-Interrupt arbeiten.

7.2.3 Digitaler Schalter (DSW)

					DSW		FNC 72		
					Digitaler Schalter				CPU
					●	●	●		
Operanden	S+	D1+	D2+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X ①	Y ②	T, C, D, V, Z	Y, M, S ③		16 Bit	32 Bit	DSW	9
					●				

- ① 8 aufeinanderfolgende Bits
- ② 4 aufeinanderfolgende Bits
- ③ n = 1 oder 2

Funktionsweise

Einlesen von BCD-Schaltern im Multiplex-Verfahren

Beschreibung

- Es können ein oder zwei (n) vierstellige BCD-Schalter in die Steuerung eingelesen werden.
- Die Anweisung kontrolliert 4 Ausgänge und 4 Eingänge. Wenn 2 vierstellige BCD-Schalter eingelesen werden, werden 8 Eingänge benötigt.
- (S+) bestimmt den ersten von vier aufeinanderfolgenden Eingängen.
- (D1+) bestimmt den ersten von vier aufeinanderfolgenden Ausgängen.
- (D2+) bestimmt den Wortoperanden, der den gelesenen Wert enthält.

HINWEISE

Die Anweisung darf nur zweimal in einem SPS-Programm eingesetzt werden.

Zur korrekten Ausführung der Anweisung muß eine Steuerung mit Transistorausgängen verwendet werden.

Beispiel ▾

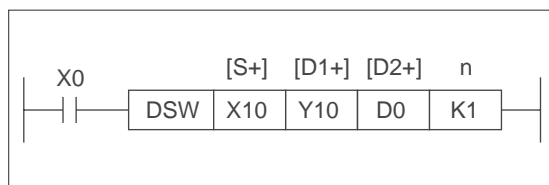


Abb. 7-8:
 Programmierbeispiel zur DSW-Anweisung

C000247C

Beispiel

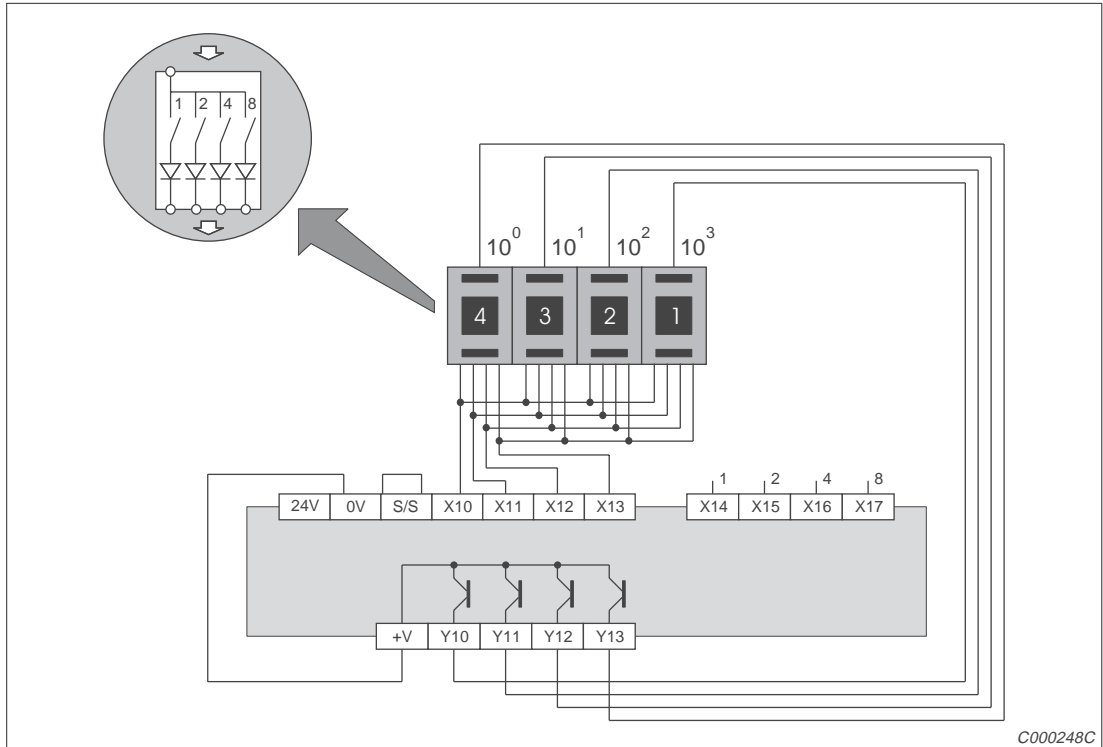


Abb. 7-9: Ein-/Ausgangsbeschaltung

Lesen des ersten Vierblocks

Die Einstellungen des vierstelligen digitalen Schalters (BCD), der mit den Eingängen X10 bis X13 verbunden ist, werden nacheinander von den Ausgängen Y10 bis Y13 gelesen und binär im Datenregister D1 gespeichert.

Die Einstellung von n ist in diesem Fall 1.

Lesen des zweiten Vierblocks

Die Einstellungen des Schalter (BCD), der mit den Eingängen X14 bis X17 verbunden ist, werden nacheinander von den Ausgängen Y10 bis Y13 gelesen und binär im Datenregister D1 gespeichert.

Die Einstellung von n ist hierbei 2.

Ist X0 eingeschaltet, arbeiten die Ausgänge Y10 bis Y13 nacheinander die Zustände der entsprechenden Eingänge X ab.

Ist ein Arbeitsvorgang abgeschlossen, wird der Sondermerker M8029 gesetzt.

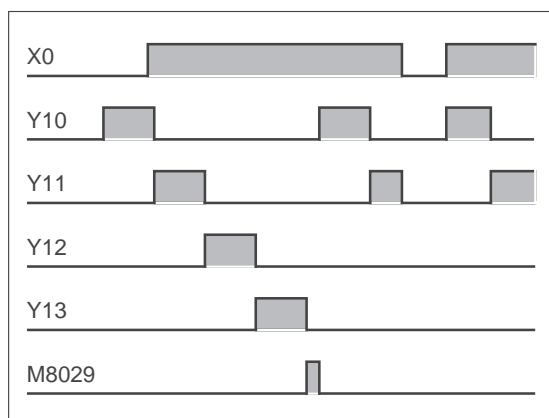



Abb. 7-10: Zeitdiagramm zum Schalten der Ausgänge

C000249C

7.2.4 7-Segment-Anzeige (SEGD)

		SEGD		FNC 73		
		7-Segment-Anzeige				
Operanden		S+	D+	Puls-Anweisung (P)	Verarbeitung	Programmschritte
		K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit 32 Bit	SEGD/ SEGDP

Funktionsweise

Ausgeben einer einstelligen Hexadezimalzahl an eine 7-Segment-Anzeige

Beschreibung

- Die hexadezimale Zahl in (S+) wird in das für eine 7-Segment-Anzeige benötigte Format gewandelt und in (D+) abgelegt.
- Die Bits b0 bis b6 von (D+) entsprechen den Segmenten der 7-Segment-Anzeige:

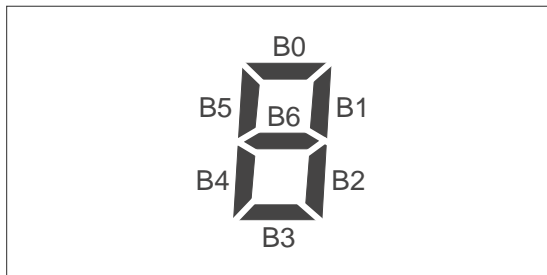


Abb. 7-11:
7-Segment-Anzeige

C000251C

Beispiel ▾

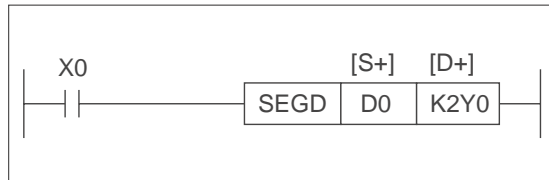


Abb. 7-12:
Programmierbeispiel zur SEGD-Anweisung

C000250C

Ausgang	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Segment	B0	B1	B2	B3	B4	B5	B6	B7

△

7.2.5 7-Segment-Anzeige mit Latch (SEGL)

				SEGL		FNC 74		
				7-Segment-Anzeige mit Latch				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	Y ①	K, H		16 Bit	32 Bit	SEGL	7
					●			

① n = 0 bis 3: 8 aufeinanderfolgende Ausgänge
 n = 4 bis 7: 12 aufeinanderfolgende Ausgänge

Funktionsweise

Ansteuern einer vierstelligen 7-Segment-Anzeige mit speichernder Anzeige

Beschreibung

- Es können bis zu zwei vierstellige 7-Segment-Anzeigen mit dieser Anweisung angesteuert werden. Die Ansteuerung erfolgt in einem Multiplex-Verfahren. Es werden 4 Taktausgänge und für jede vierstellige Anzeige nochmals 4 Datenausgänge belegt.
- Der in (S+) enthaltene Zahlenwert (max. 9999) wird in den BCD-Code gewandelt und über die Ausgänge (D+) bis ((D+)+3) ausgegeben. Wenn zwei vierstellige Anzeigen angesteuert werden sollen, erfolgt die Ausgabe der Daten für die zweite Anzeige über die Ausgänge ((D+)+10) bis ((D+)+13).
- Der jeweils an den Datenausgängen anliegende BCD-Code wird über die Taktausgänge ((D+)+4) bis ((D+)+7) der jeweiligen Stelle der Anzeige automatisch zugeordnet.
- Die Einstellung von (n) ist abhängig von vier Faktoren:
 - a) Ausgangslogik der SPS-Ausgänge (+/- -schaltend)
 - b) Logik der Datenleitungen der 7-Segment-Anzeige
 - c) Logik der Takteingänge der 7-Segment-Anzeige
 - d) Anzahl der verwendeten 7-Segment-Anzeigen

Source-Ausgang (Positive Logik)

Bei Source-Ausgängen (positiv) ist der Ausgang HIGH, wenn die interne Logik 1 lautet.

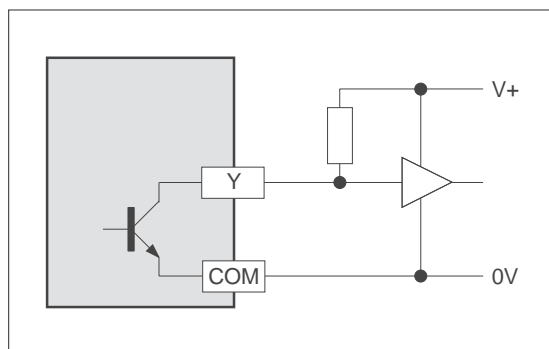


Abb. 7-13: Positive Logik

C000254C

Taktsignal-Logik: Die Daten werden gespeichert, wenn das Taktsignal HIGH ist.
 Datensignal-Logik: Aktive Datenleitungen sind HIGH.

Sink-Ausgang (Negative Logik)

Bei Sink-Ausgängen (negativ) ist der Ausgang LOW, wenn die interne Logik 1 lautet.

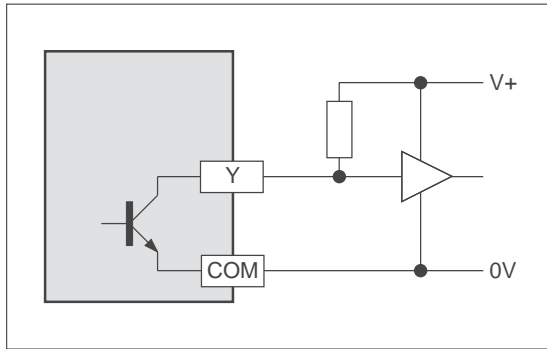


Abb. 7-14:
Negative Logik

C000255C

Taktsignal-Logik: Die Daten werden gespeichert, wenn das Taktsignal LOW ist.

Datensignal-Logik: Aktive Datenleitungen sind LOW.

SPS-Logik	Taktsignal	Datenleitung	n	
			1 Anzeige	2 Anzeigen
Positiv (+)	Positiv (HIGH)	Positiv (HIGH)	0	4
Negativ (-)	Negativ (LOW)	Negativ (LOW)		
Positiv (+)	Positiv (HIGH)	Negativ (LOW)	1	5
Negativ (-)	Negativ (LOW)	Positiv (HIGH)		
Negativ (-)	Positiv (HIGH)	Negativ (LOW)	2	6
Positiv (+)	Negativ (LOW)	Positiv (HIGH)		
Negativ (-)	Positiv (HIGH)	Positiv (HIGH)	3	7
Positiv (+)	Negativ (LOW)	Negativ (LOW)		

Tab. 7-5: 7-Segment-Anzeigelogik

HINWEISE

Zur korrekten Ausführung der Anweisung muß eine Steuerung mit Transistorausgängen verwendet werden.

Es können nur 7-Segment-Anzeigen mit Datenerhaltung verwendet werden.

Die SEGL-Anweisung kann nur zweimal im Programm eingesetzt werden.

Beispiel ▾

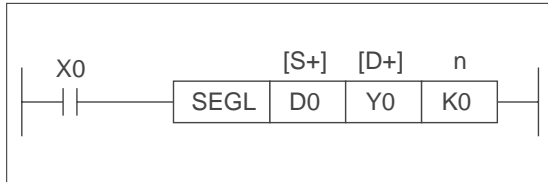
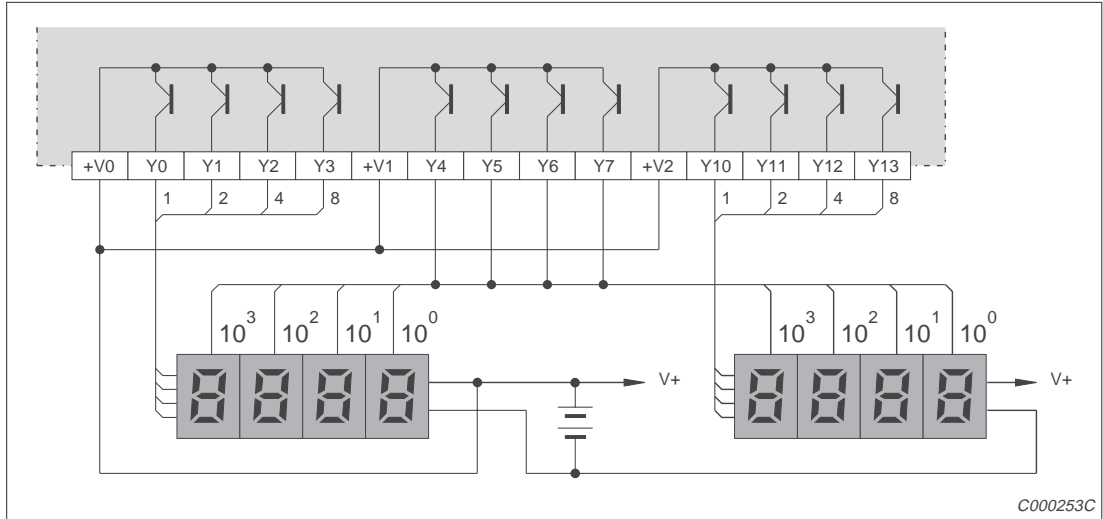


Abb. 7-15:
 Programmierbeispiel für eine SEGL-Anweisung

C000252C



C000253C

Abb. 7-16: Ausgangsbeschaltung



7.2.6 7-Segment-Anzeige mit zusätzlichen Tasten (ARWS)

					ARWS		FNC 75		
					7-Segment-Anzeige mit zusätzlichen Tasten				CPU
Operanden	S+	D1+	D2+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S ①	T, D, V, Z	Y ②	K, H n= 0 bis 3		16 Bit	32 Bit	ARWS	9

- ① 4 aufeinanderfolgende Operanden
- ② 8 aufeinanderfolgende Operanden

Funktionsweise

Auswählen und Verändern einer Stelle einer vierstelligen BCD-Zahl auf einer 7-Segment-Anzeige

Beschreibung

- Es werden vier Tasten (S+) bis ((S+)+3) abgefragt:
 - (S+) = Dekrementieren der ausgewählten Stelle
 - ((S+)+1) = Inkrementieren der ausgewählten Stelle
 - ((S+)+2) = Cursor nach rechts
 - ((S+)+3) = Cursor nach links
- Die in (D1+) abgelegten Daten werden über eine vierstellige 7-Segment-Anzeige angezeigt und mit den Tasten (S+) verändert.
- Die in (D1+) abgelegten Daten sind binäre Daten.
- Mit (D2+) und (n) werden die Ausgänge und die Art der Beschaltung festgelegt, an die die 7-Segment-Anzeige angeschlossen wird (siehe SEGL-Anweisung).

HINWEISE

Zur korrekten Ausführung der Anweisung muß eine Steuerung mit Transistorausgängen verwendet werden.

Die ARWS-Anweisung darf nur einmal im Programm eingesetzt werden.

Beispiel ▾

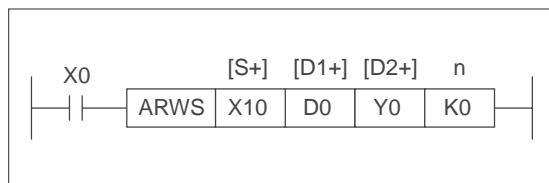


Abb. 7-17:
 Programmierbeispiel zur ARWS-Anweisung

C000256C

Beispiel

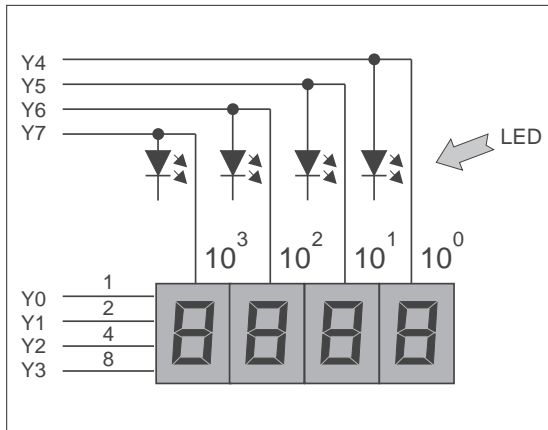


Abb. 7-18:
Beispiel für eine 7-Segment-Anzeige

C000257C

In dem 16-Bit-Datenregister D0 werden vier BCD-Werte gespeichert. Jeder BCD-Wert belegt vier Bits. Es kann maximal der Wert 9 999 in D0 dargestellt werden.

Über die Tasten bzw. Eingänge X10 bis X13 können die Position und der numerische Wert der Anzeige verändert werden (siehe Abb. 7-18).

- X11: Aufwärtszählen der ausgewählten Stelle 0-1-2-3
- X10: Abwärtszählen der ausgewählten Stelle 0-9-8-7
- X13: Verschieben nach links
- X12: Verschieben nach rechts

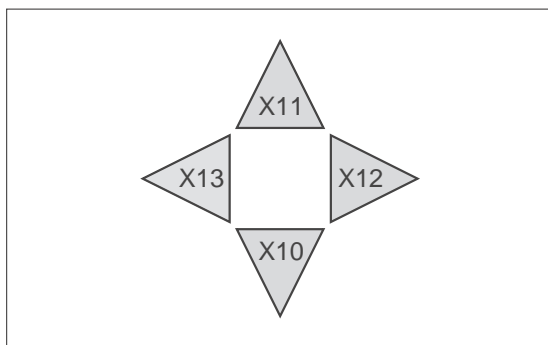


Abb. 7-19:
Beispiel für die Eingänge X10 bis X13

C000258C

Über die Eingänge X12 und X13 wird die zu verändernde Anzeigenposition festgelegt.

Ist X0 eingeschaltet, wird die Position 10^3 als Anfangsposition betrachtet.

Jedes Betätigen von X12 und X13 bewirkt, daß die Anzeigenposition in einer vorgegebenen Reihenfolge gewechselt wird:

Betätigen von X12 (Verschieben nach rechts):

$$10^3 - 10^2 - 10^1 - 10^0 - 10^3$$

Betätigen von X13 (Verschieben nach links):

$$10^3 - 10^2 - 10^1 - 10^0 - 10^3$$

Die über X12 oder X13 festgelegte Position kann von einer zusätzlichen LED in der Leitung des Strobe-Signals angezeigt werden (Y4 bis Y7).

Beispiel 


Über die Tasten bzw. Eingänge X10 und X11 wird an der festgelegten Anzeigeposition der numerische Wert verändert.

Mit X10 und X11 wird die Reihenfolge der Dateneingabe festgelegt.

Die Daten im Datenregister D0 ändern sich in der folgenden Reihenfolge:

X11: Aufwärtszählen: 0 - 1 - 2 - ... 8 - 9 - 0 - 1

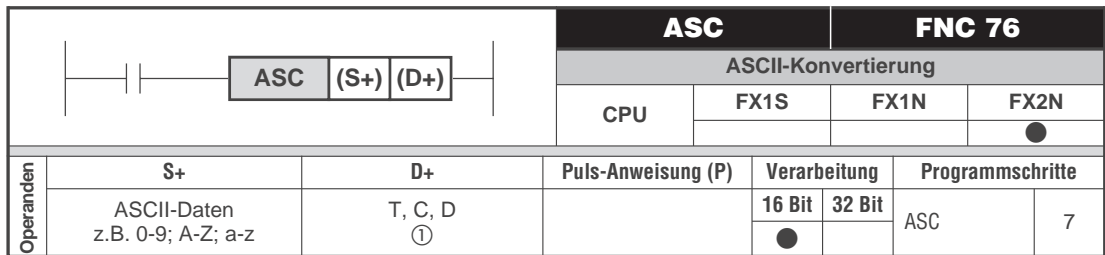
X10: Abwärtszählen: 0 - 9 - 8 - 7 - ... 1 - 0 - 9

Der aktuell gesetzte Wert wird über die 7-Segment-Anzeige dargestellt.

Mit der ARWS-Anweisung kann ein gewünschter Wert in das Datenregister D0 geschrieben und gleichzeitig auf der 7-Segment-Anzeige dargestellt werden.



7.2.7 ASCII-Konvertierung (ASC)



① 4 aufeinanderfolgende Ausgänge

Funktionsweise

Konvertieren von alphanumerischen Daten in ASCII-Daten

Beschreibung

- Die in (S+) eingegebenen alphanumerischen Daten werden in ASCII-Zeichen konvertiert und in (D+) abgespeichert.
- Die Daten in (S+) werden über ein Programmiersystem eingegeben (z.B. MELSEC ME-DOC).
- Es können maximal 8 alphanumerische Daten eingegeben werden.

Beispiel ▾

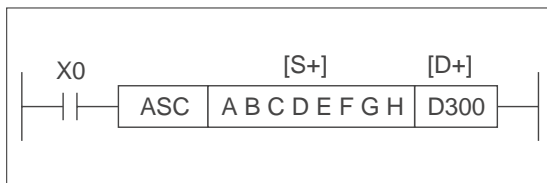


Abb. 7-20:
Programmierbeispiel einer ASC-Anweisung

C000259C

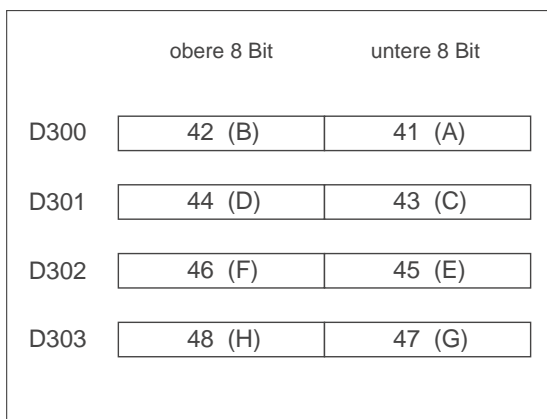


Abb. 7-21:
Speichern der Daten A bis H

C000260C



Ziffer / Buchstabe	ASCII	Ziffer / Buchstabe	ASCII	Ziffer / Buchstabe	ASCII	Ziffer / Buchstabe	ASCII
0	30	G	47	W	57	m	6D
1	31	H	48	X	58	n	6E
2	32	I	49	Y	59	o	6F
3	33	J	4A	Z	5A	p	70
4	34	K	4B	a	61	q	71
5	35	L	4C	b	62	r	72
6	36	M	4D	c	63	s	73
7	37	N	4E	d	64	t	74
8	38	O	4F	e	65	u	75
9	39	P	50	f	66	v	76
A	41	Q	51	g	67	w	77
B	42	R	52	h	68	x	78
C	43	S	53	i	69	y	79
D	44	T	54	j	6A	z	7A
E	45	U	55	k	6B		
F	46	V	56	l	6C		

Tab. 7-6: ASCII-Konvertierung

7.2.8 Datenausgabe über die Ausgänge (PR)

			PR		FNC 77		
			Datenausgabe über die Ausgänge				
	CPU	FX1S	FX1N	FX2N			
				●			
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T, C, D	Y ①		16 Bit	32 Bit	PR	5
				●			

① 10 aufeinanderfolgende Ausgänge

Funktionsweise

Ausgeben von ASCII-Zeichen über die Ausgänge

Beschreibung

- Ausgeben der ASCII-Zeichen in (S+) bis ((S+)+3) über die Ausgänge (D+).
- Die Ausgänge (D+) bis (D+ +7) stellen die Bits b0 bis b7 von (S+) dar.
- ((D+)+10) steht als Taktsignal zur Verfügung, ((D+)+11) als Ausführungs-Flag.

HINWEISE

Die PR-Anweisung darf nur zweimal im Programm verwendet werden.

Zur korrekten Ausführung der Anweisung muß eine Steuerung mit Transistorausgängen verwendet werden.

Beispiel ▾

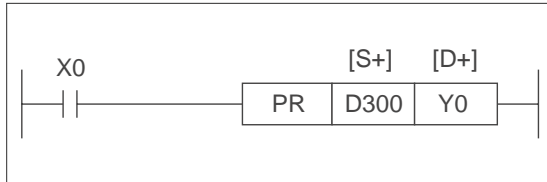


Abb. 7-22:
 Programmierbeispiel zur PR-Anweisung

C000261C

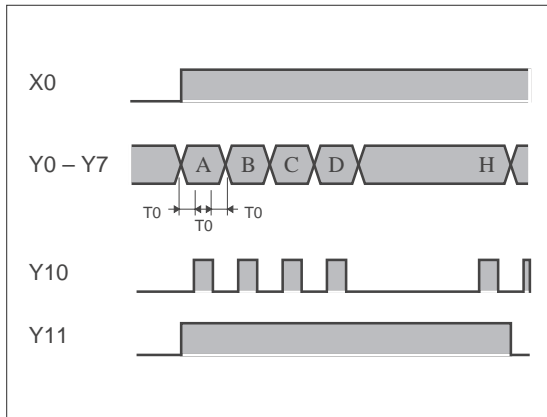


Abb. 7-23:
 Programmbeispiel zum Schalten der Ein-/Ausgänge

$T0 = \text{Zykluszeit}$

C000262C

In den Datenregistern D300 bis D303 befinden sich die ASCII-Daten der Beispielprogrammierung in Abs. 7.1.7. Es werden demnach die Buchstaben „A“ bis „H“ ausgegeben.

Als Ausgänge stehen Y0 (unteres Bit) bis Y7 (oberes Bit) sowie Y10 (Strobe-Signale) und Y11 (Ausführungs-Flag) zur Verfügung.

Ausgabeformat

Wird X0 während der Abarbeitung der Anweisung ausgeschaltet, wird die Datenübertragung eingestellt.

Der Vorgang beginnt erneut, wenn X0 wieder eingeschaltet wird.

△

7.2.9 Auslesen von Daten aus einem Sondermodul (FROM)

		FROM		FNC 78					
		Auslesen von Daten aus einem Sondermodul							
Operanden		D+	n1, n2, n3	Puls-Anweisung (P)	Verarbeitung		Programmschritte		
					16 Bit	32 Bit	FROM/FROMP	9	
		KnY, KnM, KnS, T, C, D, V, Z	K, H	●	●	●	●	DFROM/DFROMP	17

Funktionsweise

Auslesen von Daten aus den Pufferspeichern der anschließbaren Sondermodule

Beschreibung

- Auslesen von n3 Datenworten aus dem Sondermodul mit der Adresse n1.
- Es werden n3 Datenworte, ausgehend von der Pufferspeicheradresse n2 nach (D+) bis [(D+) + (n3 - 1)] geschrieben.

HINWEIS

Zulässige Bereiche: n1 = 0 bis 7
 n2 = 0 bis 31
 : n3 = 1 bis 32 bei einer 16 Bit-Anweisung
 n3 = 1 bis 16 bei einer 32 Bit-Anweisung

Jedes Sondermodul ist fortlaufend von 0 bis 7 numeriert. Die Nummerierung beginnt mit dem Modul, welches zuerst mit der SPS verbunden wird. Es können maximal 8 Sondermodule (FX1N/FX2N) an die SPS angeschlossen werden. Die Adressierung der digitalen Ein- und Ausgänge und der Sondermodule zeigt die folgende Abb..

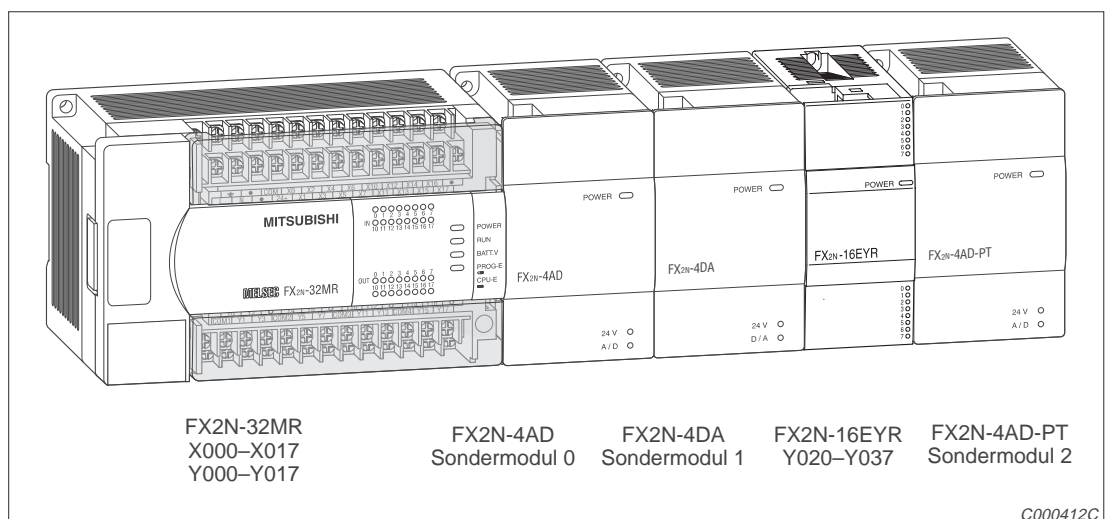


Abb. 7-24: Die Analogmodule in Verbindung mit anderen Geräten der FX-Serie

Beispiel ▾

Daten auslesen

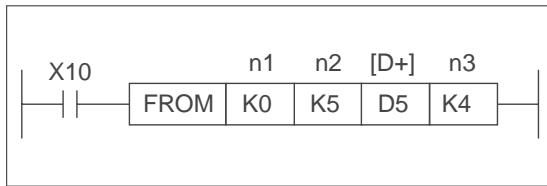


Abb. 7-25:
Programmierbeispiel zur FROM-Anweisung

C000264C

Mit dem in der Abb. gezeigten Beispiel wird der Inhalt der Pufferspeicheradressen #5 bis #8 des Sondermoduls mit der Positionsnummer 0 nach D5 bis D8 übertragen. Die Bedeutung der Adressierung ist im einzelnen in der nachfolgenden Abb. dargestellt.

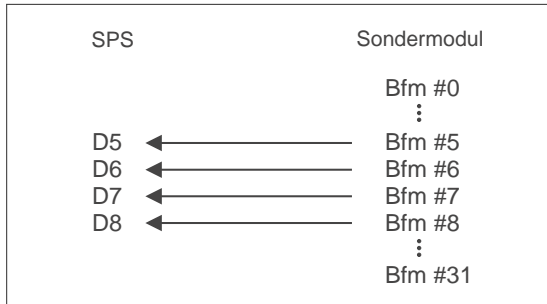


Abb. 7-26:
Adressierung bei der FROM-Anweisung

C000265C



HINWEIS

Wenn nur der Inhalt einer Pufferspeicheradresse übertragen werden soll, muß in n3 der Wert 1 eingetragen werden.

7.2.10 Schreiben von Daten in ein Sondermodul (TO)

		TO		FNC 79	
		Schreiben von Daten in ein Sondermodul			
Operanden		S+		n1, n2, n3	
		KnY, KnM, KnS, T, C, D, V, Z		K, H	
		Puls-Anweisung (P)		Verarbeitung	
				Programmschritte	
				TO/TOP	
				9	
				DTO/DTOP	
				17	

Funktionsweise

Schreiben von Daten aus der SPS in die Pufferspeicher der anschließbaren Sondermodule

Beschreibung

- Schreiben von n3 Datenworten aus der SPS in das Sondermodul mit der Adresse n1.
- Es werden n3 Datenworte, ausgehend von der Pufferspeicheradresse (S+), nach n2 bis [(n2) + (n3 - 1)] geschrieben.

HINWEIS

Zulässige Bereiche:
 n1 = 0 bis 7
 n2 = 0 bis 31
 n3 = 1 bis 32 bei einer 16 Bit-Anweisung
 n3 = 1 bis 16 bei einer 32 Bit-Anweisung

Jedes Sondermodul ist fortlaufend von 0 bis 7 nummeriert, beginnend mit dem Modul, welches zuerst mit der SPS verbunden wird. Es können maximal 8 Sondermodule an die SPS angeschlossen werden. Die Adressierung der digitalen E/As und der Sondermodule zeigt die folgende Abbildung.

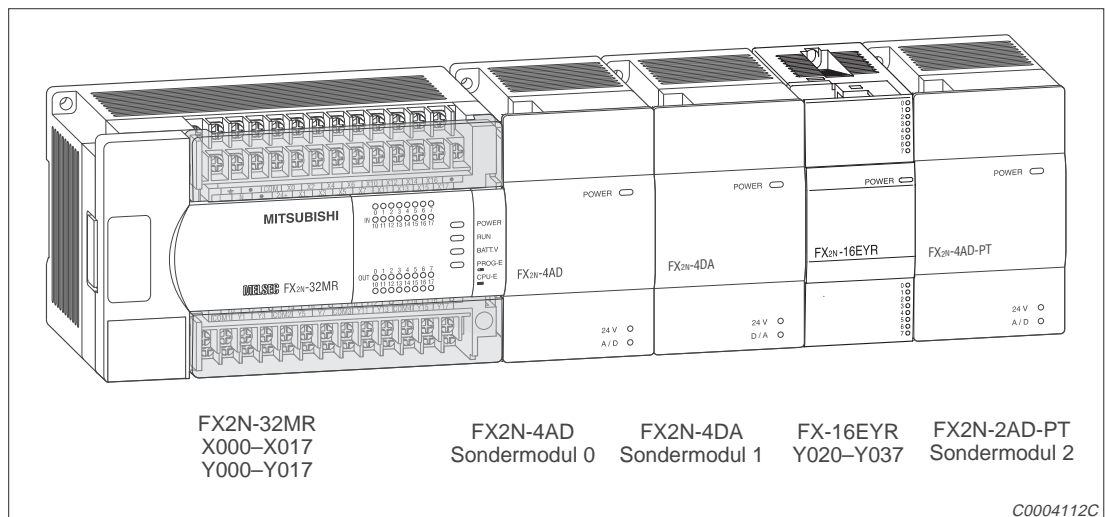


Abb. 7-27: Die Sondermodule in Verbindung mit anderen Geräten der FX-Serie

Beispiel ▾ Daten schreiben (TO)

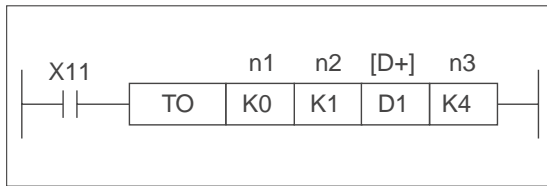


Abb. 7-28:
 Programmierbeispiel für eine TO-
 Anweisung

C000266C

Mit dem in der Abb. gezeigten Beispiel werden D1 bis D4 in die Pufferspeicheradressen #1 bis #4 des Sondermoduls mit der Positionsnummer 0 übertragen. Die Bedeutung der Adressierung ist im einzelnen wie folgt:

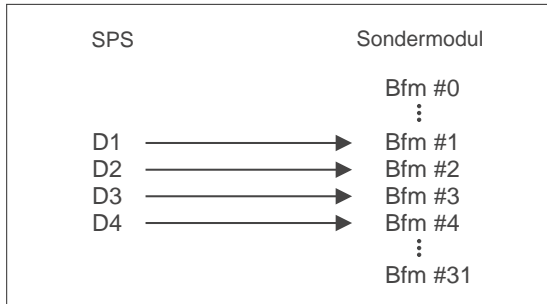


Abb. 7-29:
 Adressierung bei der TO-
 Anweisung

C000267C



HINWEIS | Wenn nur der Inhalt einer Pufferspeicheradresse übertragen werden soll, muß in n3 der Wert 1 eingetragen werden.

7.3 Serielle Kommunikation

Übersicht der Anweisungen FNC 80 bis 89

Symbol	FNC	Bedeutung	Abschnitt
RS	80	Serielle Datenübertragung	7.3.1
PRUN	81	Umlegen von Eingängen oder Merkern	7.3.2
ASCI	82	Umwandlung in ein ASCII-Zeichen	7.3.3
HEX	83	Umwandlung in einen Hexadezimalwert	7.3.4
CCD	84	Summen- und Paritätsprüfung	7.3.5
VRRD	85	Einlesen von Sollwerten vom FX1N-8AV-BD und FX2N-8AV-BD	7.3.6
VRSC	86	Einlesen von Schalterstellungen vom FX1N-8AV-BD und FX2N-8AV-BD	7.3.7
—	87		
PID	88	Regelkreisüberwachung	7.3.8
—	89		

Tab. 7-7: Übersicht der Anweisungen FNC 80 bis 89

7.3.1 Serielle Datenübertragung (RS)

				RS		FNC 80	
				Umlegen von Eingängen oder Merkern			
Operanden				CPU	FX1S	FX1N	FX2N
					●	●	●
Operanden	S+	D+	n1, n2	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	D	D	K, D		16 Bit	32 Bit	RS
					●		

Funktionsweise

Übertragung von Daten über die seriellen Kommunikationsmodule FX2N-232BD, FX232ADP, FX0N-485ADP und FX2N-485-BD.

Beschreibung

Mit Hilfe der RS-Anweisung ist das Senden und Empfangen von bzw. zu einer Vielzahl von Geräten mit serieller Schnittstelle möglich. Die Kommunikation über den seriellen Schnittstellenadapter wird dabei in vier Teilabschnitten gesteuert:

- 1 Einstellen der Kommunikationsparameter
- 2 Ausgabe der RS-Anweisung, bestehend aus:
 - (S+) = Startadresse des Übertragungspuffers
 - n1 = Länge der zu übertragenden Meldung (FX1S/FX1N max. 256 Bytes, FX2N max. 4096 Bytes)
 - (D+) = Startadresse des Empfangspuffers
 - n2 = Länge der zu empfangenden Meldung (FX1S/FX1N max. 256 Bytes, FX2N max. 4096 Bytes)

Die Summe der Meldungen beträgt bei der FX1S/FX1N max. 512 Bytes und bei der FX2N max. 8000 Bytes.

- 3 Meldung übertragen
- 4 Meldung empfangen

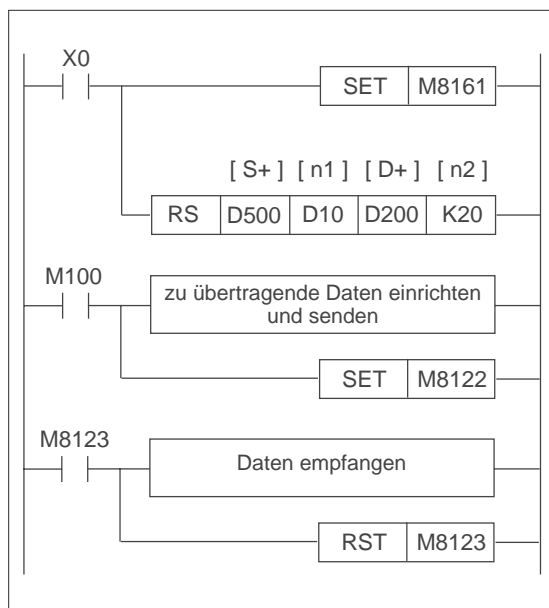


Abb. 7-30:
 Programmierbeispiel für die RS-Anweisung

C000220C

Kommunikationsparameter

Jedes Protokoll einer seriellen Kommunikation muß zunächst konfiguriert werden, damit volle Kompatibilität mit der externen Kommunikationseinheit gewährleistet ist. Das Kommunikationsprotokoll für die Module wird mit Hilfe des Sonderregisters D8120 konfiguriert. Dies ist jedoch nur bei inaktiver RS-Anweisung möglich. Die folgende Tabelle zeigt die Zusammensetzung des Sonderregisters D8120 und seiner Bedeutung für die RS232-Kommunikation.

	Beschreibung		0	1
b0	Datenlänge		7 Bit	8 Bit
b1	Parität		(00): keine Parität (01): ungerade Parität (11): gerade Parität	
b2				
b3	Stoppbit		1 Bit	2 Bit
b4	Baudrate (Bit/s)		(0011): 300, (0100): 600, (0101): 1200, (0110): 2400, (0111): 4800, (1000): 9600 (1001): 19200	
b5				
b6				
b7				
b8	Start-Byte		kein	mit D8124
b9	End-Byte		kein	mit D8125
b10	Handshake Typ1		kein	H/W1
b11	Modus Control (Line)		normal	single
b12	Handshake Typ2		kein	H/W2
b13	FX-485	Summenprüfung	keine Prüfung	Prüfung
b14	Netzwerk	Netzwerk	aus	aktiv
b15		Protokoll	Format1	Format4

Tab. 7-8:

Bits und ihre jeweilige Bedeutung für die RS-232-Kommunikation.

Das Sonderregister D8124 enthält den Wert des Start-Bytes, falls eines ausgewählt wurde. Der Grundwert ist ASCII „STX“ oder 02_H. Dieser kann jedoch vom Anwender vor Kommunikationsbeginn geändert werden.

Das Sonderregister D8125 enthält den Wert des End-Bytes, falls eines ausgewählt wurde. Der Default-Wert ist ASCII „ETX“ oder 03_H. Er kann jedoch vom Anwender vor Kommunikationsbeginn geändert werden.

Wenn die periphere Kommunikationseinheit mit Hardware-Handshake arbeitet, sollte dieser Modus gewählt werden. Ist er selektiert, dienen die DSR- und DTR-Anschlüsse (Pin 6 und 20) des Schnittstellenadapters zur Kommunikationssteuerung. Das Anschlußdiagramm finden Sie in der Hardware-Beschreibung zum Adapter. Detaillierte Informationen finden Sie im FX-Communication (User`s Manual).

Zusammensetzung der RS-Anweisung

Als Datenspeicherformat steht wahlweise der 16-Bit- oder der 8-Bit-Modus zur Verfügung. Der 16-Bit-Modus verwendet die oberen und unteren Bytes der Übertragungs- und Empfangspufferbereiche, während der 8-Bit-Modus nur die unteren 8 Bits verwendet. Gesteuert wird dies durch den Sondermerker M8161. ON steht hierbei für 8-Bit-Modus.

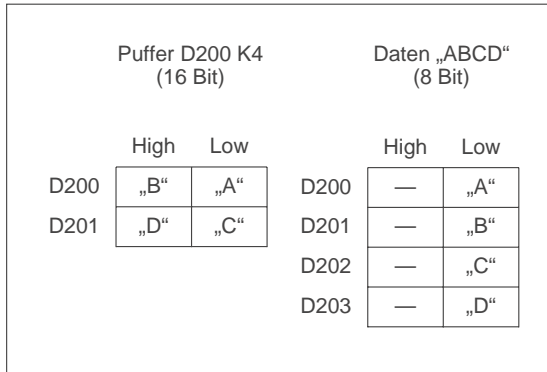


Abb. 7-31:
Adressierung bei der RS-Anweisung

Die RS-Anweisung setzt sich im einzelnen wie folgt zusammen:

- **Anweisung:**
Ist die RS-Anweisung aktiv, bedeutet das, daß eine Kommunikation möglich ist und Daten gesendet und empfangen werden können. Innerhalb eines Programms ist mehr als eine RS-Anweisung zulässig, wobei jedoch nicht mehr als eine Anweisung gleichzeitig aktiv sein darf.
- **Startadresse des Übertragungspuffers:**
Die Startadresse des Übertragungspuffers ist das erste Daten- oder File-Register (D) des Bereichs für zu übertragende Meldungen.
- **Länge der zu übertragenden Meldung:**
Hier wird die Länge der zu übertragenden Meldung festgelegt. Der Wert kann eine Konstante (K) sein; bei variierender Meldungslänge kann aber auch ein Datenregister (D) verwendet werden. Bei Verwendung eines Datenregisters läßt sich der Wert zwischen einzelnen Übertragungsanforderungen, nicht jedoch während der Übertragung selbst ändern.
- **Startadresse des Empfangspuffers:**
Die Startadresse des Empfangspuffers ist das erste Datenregister (D) des Bereichs für empfangene Meldungen.
- **Länge der zu empfangenden Meldung:**
Hier wird die maximale Länge der zu empfangenden Meldung bestimmt. Der Wert kann eine Konstante (K) sein; bei variierender Meldungslänge kann aber auch ein Datenregister (D) verwendet werden. Bei Verwendung eines Datenregisters läßt sich der Wert zwischen einzelnen Empfangsvorgängen, nicht jedoch während des Empfangs selbst ändern.

Meldung übertragen

Die Steuerung der Übertragung bzw. des Sendens einer Meldung erfolgt über den Sondermerker M8122.

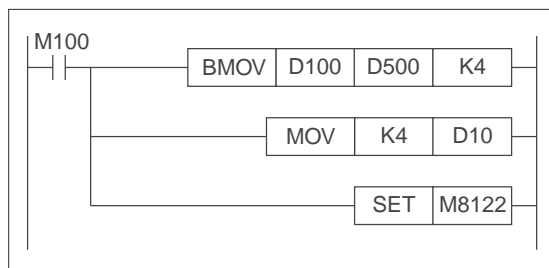


Abb. 7-32:

Programmierbeispiel zur Übertragung mit Hilfe des Sondermerkers M8122

C000175C

Zunächst müssen die zu übertragenden Daten im Übertragungspufferbereich enthalten sein. Sie können Sie auf zwei Arten dorthin übertragen:

- ① Kopieren Sie vor der Übertragung die Meldung in den Meldungspufferbereich bzw. erzeugen Sie sie mit Hilfe der MOV- oder der BMOV-Anweisung.
- ② Ändern Sie die Parameter der RS-Anweisung, um auf den entsprechenden Datenregisterbereich zuzugreifen, in dem sich die Meldung befindet. Für jede Meldung ist eine eigene RS-Anweisung erforderlich.

Im oben beschriebenen Beispiel werden die in den Datenregistern D100 bis D103 enthaltenen Daten in den Übertragungspufferbereich, beginnend bei D500, kopiert. Danach wird die Meldungslänge auf 8 Byte festgelegt, indem der Wert für die Länge der zu übertragenden Meldung mit Hilfe des Datenregisters D10 geändert wird.

Sind die Daten definiert und korrekt lokalisiert, kann der Übertragungsmerker M8122 auf ON gesetzt werden. Nun beginnt der Versand der Daten, und der Merker M8122 wird, sobald die Datenübertragung abgeschlossen ist, automatisch zurückgesetzt.

Es wird empfohlen, den Merker mit einem Impuls-Signal zu setzen, da der Merker andernfalls nach erfolgter Übertragung wieder auf ON gesetzt wird und sich die Datenübertragung wiederholt.

Bei Verwendung von Start- und/oder Endsignalen (Header/Terminator) werden diese automatisch der Meldung vor der Übertragung hinzugefügt.

Ein Übertragungszähler über das Sonderregister ist ebenfalls möglich. Das Sonderregister D8122 kann während des Sendevorgangs überprüft werden, so daß Sie den Verlauf der Übertragung verfolgen können. Der Wert in D8122 beginnt bei der vollen Meldungslänge und wird pro übertragenem Daten-Byte um jeweils eins heruntergezählt.

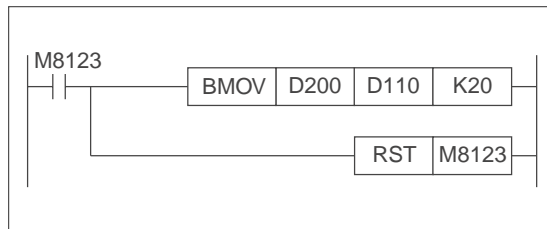
HINWEIS

Die Start- und Endsignale werden im D8122 nicht mitgezählt.

Meldung empfangen

Die RS-Anweisung steuert automatisch den Empfang einer Meldung. Sobald eine Meldung vollständig empfangen ist, werden die Daten im Empfangspufferbereich gespeichert, und der Sondermerker M8123 wird auf ON gesetzt.

Eventuell verwendete Start- und/oder Endsignale werden automatisch entfernt, bevor die Meldung im Pufferbereich abgelegt wird.

**Abb. 7-33:**

Programmierbeispiel zur Übertragung mit Hilfe des Sondermerkers M8123

C000176C

Sobald der Sondermerker auf ON steht, sollten die im Empfangspuffer enthaltenen Daten weiter verarbeitet und der Merker auf OFF zurückgesetzt werden, damit der Bereich zum Empfang von weiteren Meldungen zur Verfügung steht. Der Merker wird automatisch zurückgesetzt, wenn die RS-Anweisung abgeschaltet wird.

Im oben beschriebenen Programmbeispiel wird Sondermerker M8123 „Meldung empfangen“ überprüft. Wird er aktiviert, werden alle im Empfangspuffer enthaltenen Daten in eine andere Adresse kopiert, und der Empfangsmerker wird zurückgesetzt. Nachdem der Empfangspufferbereich wieder frei ist, können die empfangenen Daten nach Bedarf weiter verarbeitet werden.

HINWEIS

Das gleichzeitige Senden und Empfangen von Daten ist nicht möglich. Der Merker M8121 (Empfangskennung) steht während des Empfangsvorgangs auf ON. Zwar kann der Merker M8122 (Sendekennung) zur selben Zeit auch auf ON stehen, aber der tatsächliche Sendevorgang wird so lange verzögert, bis die Meldung vollständig empfangen ist.

Die Einrichtung eines Empfangszähler ist ebenfalls möglich. Während des Empfangs kann im Sonderregister D8123 überprüft werden, wieviel Bytes aktuell übertragen worden sind. Nach vollständig empfangener Meldung wird die Gesamtlänge der Meldung angezeigt.

Start- und Endsignale (Headers, Terminators)

● Beschreibung

Bei der Datenkommunikation ist es oft erforderlich, den Anfang und das Ende einer Meldung besonders zu kennzeichnen. Dies geschieht üblicherweise mit Hilfe bestimmter Zusätze zur Meldung, sogenannten Start- und Endsignalen. Mit der RS-Anweisung haben Sie die Möglichkeit, der Meldung automatisch ein Start-Byte und/oder ein End-Byte hinzuzufügen.

Ausgewählt werden das Start- und das End-Byte durch Setzen der Bits b8 und b9 im Datenregister D8120 der Kommunikationsparameter.

● Während der Übertragung

Wurde ein Start-Signal gewählt, wird das untere Byte des Sonderregisters D8124 als das erste Byte jeder übertragenen Meldung gesendet.

Wurde ein End-Signal gewählt, wird das untere Byte des Sonderregisters D8125 als das letzte Byte jeder übertragenen Meldung gesendet.

● Während des Empfangs

Wurde ein Start-Signal gewählt, werden alle empfangenen Daten so lange ignoriert, bis das Start-Byte empfangen wurde. Wurde kein Start-Signal gewählt, wird das erste empfangene Byte als Meldungsinhalt gelesen.

Wurde ein End-Signal gewählt, werden im Zuge des Lesevorgangs alle empfangenen Daten als Meldung gelesen, bis das End-Signal empfangen wird oder die Gesamtmeldungslänge erreicht ist, d.h. der Empfangspuffer voll ist.

Wurde kein End-Signal gewählt, dauert der Lesevorgang so lange an, bis der Empfangspuffer voll ist, d.h. die Meldung muß in voller Länge eingegangen sein, bevor sie als vollständig gewertet wird.

Nach Empfang einer vollständigen Meldung wird der Merker M8123 gesetzt. Alle danach empfangenen Daten werden so lange ignoriert, bis dieser Empfangsmerker wieder gelöscht wird.

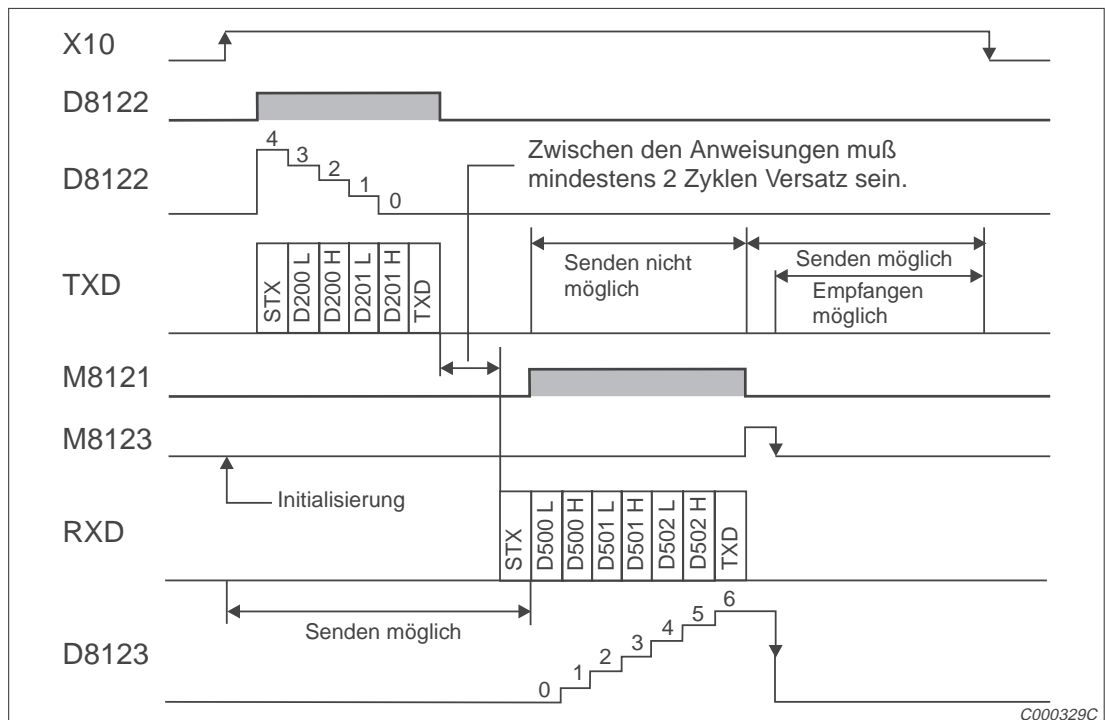


Abb. 7-34: Darstellung des zeitlichen Ablaufs

7.3.2 Umlegen von Eingängen oder Merkern (PRUN)

		PRUN		FNC 81	
		Umlegen von Eingängen oder Merkern			
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung	
	KnX, KnM n = 1 – 8	KnM, KnY n = 1 – 8	●	16 Bit ●	32 Bit ●
				Programmschritte	
				PRUN/PRUNP	5
				DPRUN/DPRUNP	9

Funktionsweise

Umlegen von Eingängen oder Merkern auf den im Parallelbetrieb automatisch übertragenen Datenbereich

Beschreibung

- Die PRUN-Anweisung unterscheidet sich von der MOV-Anweisung darin, dass sie oktalt arbeitet.
- Da die Übertragung oktalt erfolgt, sollte (S+) mit X / M 0, 10, 20, 30 ... beginnen.

HINWEIS | Der Aufbau und die Initialisierung einer parallelen Datenverarbeitung wird im Anhang dieses Handbuches näher erläutert.

Beispiel ▾

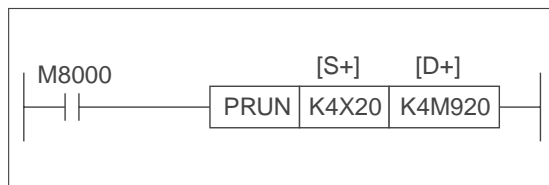
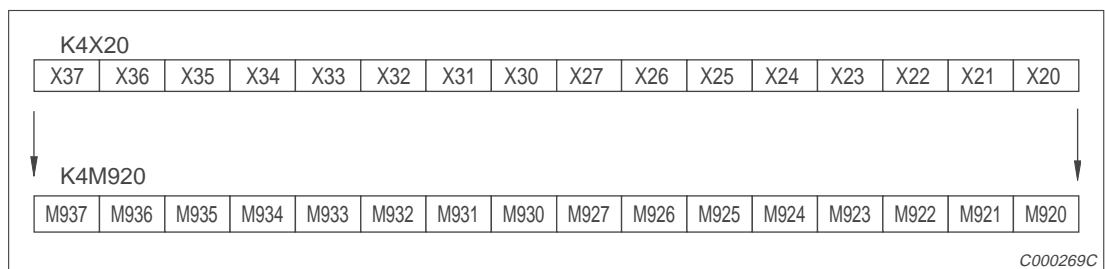


Abb. 7-35:
Programmierbeispiel zur PRUN-Anweisung

C000268C



C000269C

Abb. 7-36: Zuordnung der Merker

Die Merker M929 und M928 werden nicht beschrieben, da die PRUN-Anweisung oktalt arbeitet△

7.3.3 ASCII-Umwandlung (ASCII)

				ASCII		FNC 82		
				Umwandlung in ein ASCII-Zeichen				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, T, C, D, KnX, KnY, KnM, KnS	T, C, D, KnY, KnM, KnS	K, H	●	16 Bit	32 Bit	ASCII/ASCIP	7

Funktionsweise

Umwandlung eines hexadezimalen Wertes in ein ASCII-Zeichen

Beschreibung

Die ASCII-Anweisung ermöglicht die Umwandlung eines hexadezimalen Wertes aus einem Datenregister in ein ASCII-Zeichen.

Die ASCII-Anweisung setzt sich im einzelnen wie folgt zusammen:

- der Startadresse (S+), wo die hexadezimalen Daten gespeichert sind
- die Zieladresse (D+), wo die umgewandelten ASCII-Zeichen gespeichert werden sollen
- die Angabe über die Anzahl der Zeichen (n), d.h. die Zahl der in ASCII-Zeichen umzuwandelnden hexadezimalen Ziffern

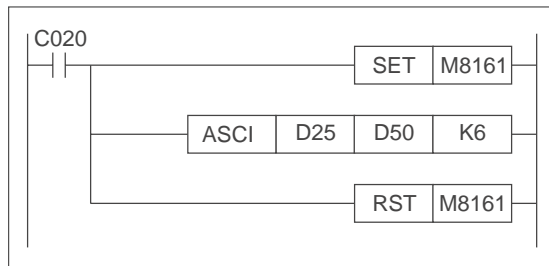


Abb. 7-37:

Programmierbeispiel für die ASCII-Anweisung

C000177C

Das dargestellte Programm führt die Umwandlung aus, wenn der Zähler C20 seinen festgelegten Wert erreicht. Während die ASCII-Anweisung aktiv ist, werden die sechs (K6) hexadezimalen Ziffern aus den Datenregistern D25 und D26 in ASCII-Zeichen umgewandelt und in den Datenregistern D50 bis D55 (8-Bit-Format) gespeichert. Dabei ist jedes Zeichen ein Byte.

Als Datenspeicherformat steht wahlweise der 16-Bit- oder der 8-Bit-Modus zur Verfügung. Der 16-Bit-Modus verwendet die oberen und unteren Bytes der Quelleinheiten, während der 8-Bit-Modus nur die unteren 8 Bits verwendet. Gesteuert wird dies durch den Sondermarker M8161. ON steht hierbei für 8-Bit-Modus.

HINWEIS

Die Anweisung „SET M8161“ ist nur dann erforderlich, wenn der 8-Bit-Modus verwendet werden soll.

Zusammensetzung der ASCII-Anweisung

- **Startadresse**
Der hier definierte Wert bezeichnet den ersten Wortoperanden, der die umzuwandelnden hexadezimalen Zahlen enthält. Sollen mehr als 4 Ziffern umgewandelt werden, werden die folgenden Wortoperanden ebenfalls gelesen, bis alle gewünschten Ziffern umgewandelt sind.
- **Zieladresse**
Der hier definierte Wert bezeichnet den ersten Wortoperanden, der die ASCII-Zeichen enthalten soll. Jeder Wortoperand kann 2 Zeichen (2 Bytes) enthalten. Die der Startadresse folgenden Wortoperanden werden so lange verwendet, bis alle Zeichen gespeichert sind.
- **Anzahl der Zeichen**
Der hier angegebene Wert kann nur ein Dezimalwert (K) oder ein hexadezimaler Wert (H) sein. Er bezeichnet die Anzahl der hexadezimalen Ziffern, die umgewandelt werden sollen, und wieviele ASCII-Zeichen gespeichert werden sollen. Die Anzahl der Zeichen kann zwischen 1 und 256 Ziffern betragen.
- **Rücksetzung des Datenspeicherformats**
Die Funktion „RST M8161“ setzt das Datenspeicherformat auf den Ausgangswert von 16 Bit zurück.

HINWEIS

Die Funktion „RST M8161“ wird nur dann benötigt, wenn in der ASCII-Anweisung das 8-Bit-Datenspeicherformat verwendet wird, während andere Anweisungen in Ihrem Programm das 16-Bit-Datenspeicherformat verwenden.

Anwendungsbeispiel

Entsprechend dem Programmbeispiel in der Abb. 7-36 zeigt das folgende Diagramm die Ergebnisse sowohl für das 16-Bit- als auch das 8-Bit-Format.

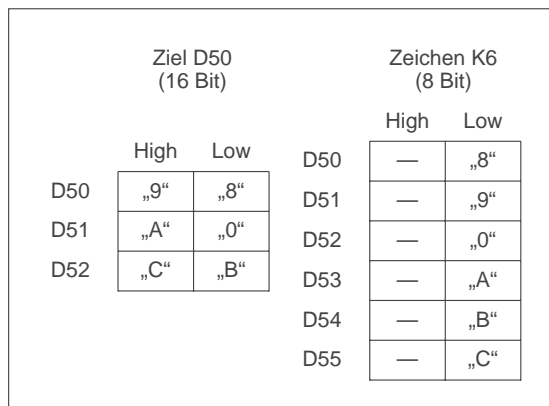


Abb. 7-38:
Grafische Darstellung im 16-Bit- und im 8-Bit-Format

Die folgende Tabelle enthält die ASCII-Codes für alle wandelbaren hexadezimalen Zahlen sowohl in hexadezimalen als auch in dezimalen Ziffern.

HEX	ASCII		Symb.	HEX	ASCII		Symb.	HEX	ASCII		Symb.	HEX	ASCII		Symb.
	HEX	DEZI			HEX	DEZI			HEX	DEZI			HEX	DEZI	
0	30	48	„0“	4	34	52	„4“	8	38	56	„8“	C	43	67	„C“
1	31	49	„1“	5	35	53	„5“	9	39	57	„9“	D	44	68	„D“
2	32	50	„2“	6	36	54	„6“	A	41	65	„A“	E	45	69	„E“
3	33	51	„3“	7	37	55	„7“	B	42	66	„B“	F	46	70	„F“

Tab. 7-9: ASCII-Code-Tabelle

7.3.4 Hexadezimal-Umwandlung (HEX)

				HEX		FNC 83		
				Umwandlung in einen Hexadezimalwert				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, T, C, D, KnX, KnY, KnM, KnS	T, C, D, KnY, KnM, KnS	K, H	●	16 Bit	32 Bit	HEX/HEXP	7

Funktionsweise

Umwandlung eines ASCII-Zeichens in einen Hexadezimalwert

Beschreibung

Die HEX-Anweisung ermöglicht die Umwandlung eines ASCII-Zeichens aus einem Datenregister in einen hexadezimalen Wert.

Die HEX-Anweisung setzt sich im einzelnen wie folgt zusammen:

- der Startadresse (S+), wo die ASCII-Zeichen gespeichert sind
- die Zieladresse (D+), wo die umgewandelten hexadezimalen Daten gespeichert werden sollen
- die Angabe über die Anzahl der Zeichen (n), d.h. die Zahl der in hexadezimale Ziffern umzuwandelnden ASCII-Zeichen

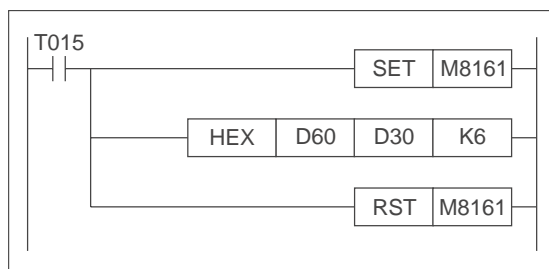


Abb. 7-39: Programmierbeispiel für die HEX-Anweisung

C000178C

Das in Abb. 7-38 dargestellte Programm führt die Umwandlung aus, wenn der Zähler T15 seinen festgelegten Wert erreicht. Während die HEX-Anweisung aktiv ist, werden die sechs (K6) ASCII-Zeichen aus den Datenregistern D60 und D65 in hexadezimale Ziffern umgewandelt und in den Datenregistern D30 und D31 (8-Bit-Format) gespeichert. Dabei ist jede Ziffer ein Byte.

Als Datenspeicherformat steht wahlweise der 16-Bit- oder der 8-Bit-Modus zur Verfügung. Der 16-Bit-Modus verwendet die oberen und unteren Bytes der Quelleinheiten, während der 8-Bit-Modus nur die unteren 8 Bits verwendet. Gesteuert wird dies durch den Sondermerker M8161. ON steht hierbei für 8-Bit-Modus.

HINWEIS

Die Anweisung „SET M8161“ ist nur dann erforderlich, wenn der 8-Bit-Modus verwendet werden soll.

Zusammensetzung der HEX-Anweisung

- **Startadresse**
Der hier definierte Wert bezeichnet den ersten Wortoperanden, der die umzuwandelnden ASCII-Zeichen enthält. Sollen mehr als 2 Zeichen (2 Bytes) umgewandelt werden, werden die folgenden Wortoperanden ebenfalls gelesen, bis alle gewünschten Zeichen umgewandelt sind.
- **Zieladresse**
Der hier definierte Wert bezeichnet den ersten Wortoperanden, der die hexadezimalen Zahlen enthalten soll. Jeder Wortoperand kann 4 Ziffern enthalten. Die der Startadresse folgenden Wortoperanden werden so lange verwendet, bis alle Ziffern gespeichert sind.
- **Anzahl der Zeichen**
Der hier angegebene Wert kann nur ein Dezimalwert (K) oder ein hexadezimaler Wert (H) sein. Er bezeichnet die Anzahl der ASCII-Zeichen, die umgewandelt werden sollen, und wieviele hexadezimale Ziffern gespeichert werden sollen. Die Anzahl der Zeichen kann zwischen 1 und 256 Ziffern betragen.
- **Rücksetzung des Datenspeicherformats**
Die Funktion RST M8161 setzt das Datenspeicherformat auf den Default-Wert von 16 Bit zurück.

HINWEIS

Die Funktion „RST M8161“ wird nur dann benötigt, wenn in dieser Anweisung das 8-Bit-Datenspeicherformat verwendet wird, während andere Anweisungen in Ihrem Programm das 16-Bit-Datenspeicherformat verwenden.

Anwendungsbeispiel

Entsprechend dem Programmbeispiel in Abb. 7-38 zeigt das folgende Diagramm die Ergebnisse sowohl für das 16-Bit- als auch das 8-Bit-Format.

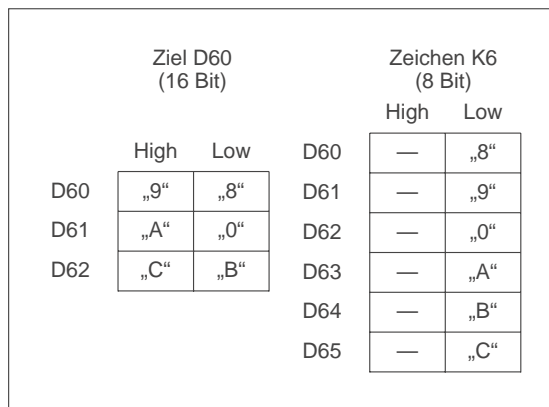


Abb. 7-40:
Grafische Darstellung im 16-Bit- und im 8-Bit-Format

Die folgende Tabelle enthält die ASCII-Codes für alle wandelbaren hexadezimalen Zahlen sowohl in hexadezimalen als auch in dezimalen Ziffern.

HEX	ASCII		Symb.	HEX	ASCII		Symb.	HEX	ASCII		Symb.	HEX	ASCII		Symb.
	HEX	DEZI			HEX	DEZI			HEX	DEZI			HEX	DEZI	
0	30	48	„0“	4	34	52	„4“	8	38	56	„8“	C	43	67	„C“
1	31	49	„1“	5	35	53	„5“	9	39	57	„9“	D	44	68	„D“
2	32	50	„2“	6	36	54	„6“	A	41	65	„A“	E	45	69	„E“
3	33	51	„3“	7	37	55	„7“	B	42	66	„B“	F	46	70	„F“

Tab. 7-10: ASCII-Code-Tabelle

7.3.5 Summen- und Paritätsprüfung (CCD)

				CCD		FNC 84		
				Summen- und Paritätsprüfung				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S+	D+	n	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T, C, D, KnX, KnY, KnM, KnS	T, C, D, KnY, KnM, KnS	K, H			16 Bit	32 Bit	CCD/CCDP
					●			

Funktionsweise

Berechnung einer Summenprüfung und einer Paritätskontrolle

Beschreibung

Die CCD-Anweisung ermöglicht eine Berechnung einer Summenprüfung und einer Paritätskontrolle eines Datenbereiches.

Die CCD-Anweisung setzt sich im einzelnen wie folgt zusammen:

- der Startadresse (S+), wo die Daten gespeichert sind
- die Zieladresse (D+), wo der Wert der Summenprüfung gespeichert werden soll
- die Angabe über die Anzahl der Zeichen (n), d.h. die Byte-Zahl der zu prüfenden Daten

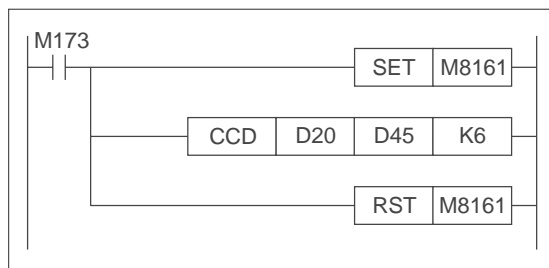


Abb. 7-41:
 Programmierschema für die CCD-Anweisung

C000180C


Das in Abb. 7-40 dargestellte Programm führt die Summen- und Paritätsprüfung aus, wenn der Merker M173 aktiviert wird. Während die CCD-Anweisung aktiv ist, werden sechs (K6) Daten-Bytes aus den Datenregistern D20 bis D25 (8-Bit-Format) summiert und der summierte Wert und die Paritätsprüfung in den Datenregistern D45 bzw. D46 gespeichert.

Als Datenspeicherformat steht wahlweise der 16-Bit- oder der 8-Bit-Modus zur Verfügung. Der 16-Bit-Modus verwendet die oberen und unteren Bytes der Quelleinheiten, während der 8-Bit-Modus nur die unteren 8 Bits verwendet. Gesteuert wird dies durch den Sondermerker M8161. ON steht hierbei für 8-Bit-Modus.

HINWEIS

Die Anweisung „SET M8161“ ist nur dann erforderlich, wenn der 8-Bit-Modus verwendet werden soll.

7.3.6 Einlesen von Sollwerten vom FX1N-8AV-BD und FX2N-8AV-BD (VRRD)

		VRRD		FNC 85	
		Einlesen von Sollwerten vom FX-8AV			
Operanden		CPU	FX1S	FX1N	FX2N
			●	●	●
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung	Programmschritte
	K, H S = 0 bis 7	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit ●	32 Bit VRRD/ VRRDP

Funktionsweise

Einlesen der auf einem FX1N-8AV-BD und FX2N-8AV-BD eingestellten Sollwerte in die FX-Steuerung

Beschreibung

Mit der Anweisung VRRD wird die Einstellung des Potentiometers (S+) in einen 8-Bit-Wert gewandelt und in (D+) abgespeichert.

HINWEISE

Die Potentiometer sind von 0 bis 7 nummeriert.

Die Module FX1N-8AV-BD und FX2N-8AV-BD werden in den Erweiterungssteckplatz der FX1S/FX1N- bzw. FX2N-CPU eingesetzt.

Beispiel ▾

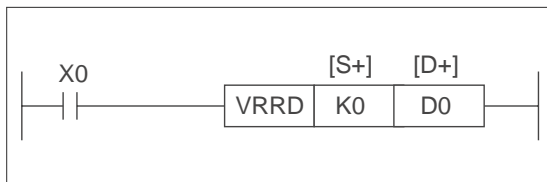
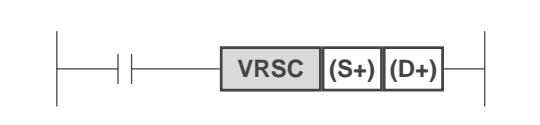


Abb. 7-43:
Programmierbeispiel zur VRRD-Anweisung

C000270C

Das Potentiometer „0“ wird eingelesen, und der eingestellte Wert wird in dem Datenregister D0 abgespeichert. △

7.3.7 Einlesen von Schalterstellungen von FX1N-8AV-BD und FX2N-8AV-BD (VRSC)

		VRSC		FNC 86		
		Einlesen von Schalterstellungen von FX-8AV				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte
	K, H S = 0 – 7	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit ●	32 Bit	VRSC/ VRSCP

Funktionsweise

Einlesen der auf einem FX1N-8AV-BD und FX2N-8AV-BD eingestellten Schaltwerte in die FX-Steuerung

Beschreibung

- Mit der Anweisung VRSC wird die Schaltstellung (0 bis 10) des Potentiometers (S+) nach (D+) gelesen.
- Die eingestellten Werte werden auf ganze Zahlen gerundet.

HINWEISE

Die Potentiometer sind von 0 bis 7 nummeriert.

Die Module FX1N-8AV-BD und FX2N-8AV-BD werden in den Erweiterungssteckplatz der FX1S/FX1N- bzw. FX2N-CPU eingesetzt.

Beispiel ▾

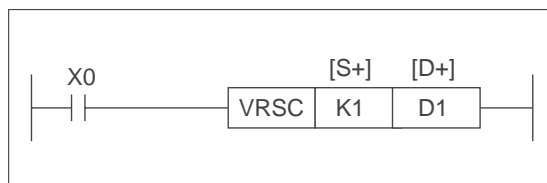


Abb. 7-44:

Programmierbeispiel zur VRSC-Anweisung

C000271C

Die Schaltstellung von Schalter „1“ wird in das Datenregister D1 geschrieben.

7.3.8 Programmierung eines geschlossenen Regelkreises (PID)

				PID		FNC 88																	
				Regelkreisüberwachung																			
<table border="1"> <tr> <th rowspan="2">Operanden</th> <th>S1+, S2+</th> <th>S3+</th> <th>D+</th> <th rowspan="2">Puls-Anweisung (P)</th> <th colspan="2">Verarbeitung</th> <th colspan="2">Programmschritte</th> </tr> <tr> <td>D</td> <td>D^②</td> <td>D</td> <td>16 Bit</td> <td>32 Bit</td> <td>PID</td> <td>9</td> </tr> </table>				Operanden	S1+, S2+	S3+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte		D	D ^②	D	16 Bit	32 Bit	PID	9	CPU	FX1S	FX1N	FX2N
					Operanden	S1+, S2+	S3+		D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte										
D	D ^②	D	16 Bit	32 Bit		PID	9																
				● ^①	●	●																	
				●																			

① Die Steuerungen der FX1S-Serie können keine Analogwerte einlesen. Die Daten für die Regelung müssen der SPS über eine serielle Verbindung zugeführt werden.

② D0 bis D975 können gewählt werden

Funktionsweise

Programmierung eines geschlossenen Regelkreises unter Angabe von 25 Regelparametern

Beschreibung

- Mit der PID-Anweisung kann die Regelung eines Istwertes in einem geschlossenen Regelkreis erfolgen.
- (S1+) gibt den gewünschten Sollwert für den Regelkreis an.
- (S2+) liest den erfaßten Istwert als Rückmeldung für den Regler ein.
- (S3+) ist die Anfangsadresse des Registerbereichs, in dem die Regelkreisparameter gespeichert werden. Hierfür stehen 25 aufeinander folgende Datenregister zur Verfügung. Tabelle 7-11 enthält eine Übersicht der Parameter.
- Nach (D+) wird der berechnete Ausgangswert, der an den Regelprozeß ausgegeben wird, geschrieben.

Die PID-Anweisung arbeitet mit der folgenden mathematischen Formel zur Berechnung des Ausgangswertes:

$$\text{Ausgangswert} = K_p \left\{ \varepsilon + K_D T_D \frac{d\varepsilon}{dt} + \frac{1}{T_I} \int \varepsilon dt \right\}$$

Hierfür gilt:

- K_p = Proportionalfaktor
- ε = Abweichung
- K_D = Differentialfaktor
- T_D = differentielle Zeitkonstante
- T_I = Integrierzeitkonstante

(S3+) Parameter Nr.	Bezeichnung	Beschreibung	Wertebereich
+0	Sampling-Zeit	Abtastintervall für Prozess-Istwert	1 – 32767 ms
+1	Bewegungsrichtung/ Alarmkontrolle	Bit 0: 0 = Vorwärts; 1 = Rückwärts	—
		Bit 1: 0/1 = Istwert-Alarmmeldung AUS/EIN	
		Bit 2: 0/1 = Ausgangswert-Alarmmeldung AUS/EIN	
		Bit 3: Reserviert	
		Bit 4: Auto-Tuning-Funktion ^① 1 = Start; 0 = Aus	
		Bit 5: Begrenzung des Ausgangswertes (FX2N) 1 = Aktiviert; 0 = deaktiviert	
		Bild 6 - 15: Reserviert	
+2	Input-Filter (α)	Einstellwert für Input-Filter	0 – 99 %
+3	Proportionalfaktor (K_p)	Multiplikationsfaktor für Proportionalregelung	1 – 32767 %
+4	Integrierzeitkonstante (T_I)	Faktor für Kehrwertmultiplikation bei Integralregelung. Die Auswahl des Werts Null verhindert die Integralregelung.	0 – 32767 x 100 ms
+5	Differentialfaktor (K_D)	Multiplikationsfaktor für Differentialregelung	0 – 100 %
+6	Differentielle Zeitkonstante (T_D)	Multiplikationsfaktor für Differentialregelung. Die Auswahl des Werts Null verhindert die Differentialregelung.	0 – 32767 x 10 ms
+7 – +19	Reserviert	—	—
+20	Kontrollwert für Istwert-Alarmmeldung (ansteigend)	Alarmausgabe, wenn der Istwert diesen Kontrollwert übersteigt.	0 – 32767
+21	Kontrollwert für Istwert-Alarmmeldung (abfallend)	Alarmausgabe, wenn der Istwert diesen Kontrollwert unterschreitet.	0 – 32767
+22	Kontrollwert für Ausgangswert-Alarmmeldung (ansteigend)	Alarmausgabe, wenn der Ausgangswert diesen Kontrollwert übersteigt.	0 – 32767
	Obere Begrenzung des Ausgangswertes (FX2N)	Vom Anwender festgelegte obere Begrenzung des Ausgangswertes (D+). (Aktiv, wenn Bit 5 von (S3+)+1 gesetzt ist)	-32768 – 32767
+23	Kontrollwert für Ausgangswert-Alarmmeldung (abfallend)	Alarmausgabe, wenn der Ausgangswert diesen Kontrollwert unterschreitet.	0 – 32767
	Untere Begrenzung der Ausgangswertes (FX2N)	Vom Anwender festgelegte untere Begrenzung des Ausgangswertes (D+). (Aktiv, wenn Bit 5 von (S3+)+1 gesetzt ist)	-32768 – 32767
+24	Alarmausgabe	Bit 0: Istwertalarm (Überschreitung)	—
		Bit 1: Istwertalarm (Unterschreitung)	
		Bit 2: Ausgangswertalarm (Überschreitung)	
		Bit 3: Ausgangswertalarm (Unterschreitung)	

Tab. 7-11: Übersicht der Regelparameter

- ① Die FX1S/FX1N- und die FX2N-Serie verfügen über eine Autotuning-Funktion. Diese Funktion bestimmt die Startwerte der Regelparameter K_p ((S3+)+3), T_I ((S3+)+4), T_D ((S3+)+6) und die Richtung der Regelung ((S3+)+1), Bit (0). Alle anderen Parameter müssen vom Anwender angegeben werden.
Mit Setzen des Bits 4 in ((S3+)+1) wird die Auto-Tuning-Funktion aktiviert.
Der Ausgangswert (MV) wird im Bereich des angegebenen Startwertes gehalten, und die Antwort des zu regelnden Systems (PV) wird überwacht. Erreicht dieser Wert $\frac{1}{3}$ des Sollwertes (SV), wird die Auto-Tuning-Funktion abgeschaltet und Bit 4 von ((S3+)+1) zurückgesetzt.

Folgende Punkte sind bei der Verwendung der Auto-Tuning-Funktion zu beachten

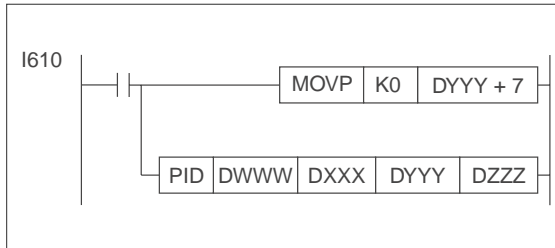
- Der Unterschied zwischen Ist-Wert (CV) und Soll-Wert (SV) muss 150 % betragen.
- Die Abtastzeit muss ein Vielfaches der Zykluszeit und größer 1 Sekunde sein.
- Vor dem Starten der Autotuning-Funktion muss das System stabil sein.

Die Parameterwerte können direkt in die Datenregister geschrieben werden. Sollen die Daten bei Abschaltung der Versorgungsspannung erhalten bleiben, müssen Sie die batteriegepufferten Datenregister verwenden. Eine andere Möglichkeit ist, die Parameterdaten in File-Registern abzulegen und über die BMOV-Anweisung (BMOV, FNC 15) in die gewünschten Datenregister zu schreiben. Dies hat den Vorteil, dass Sie mehrere Sätze mit Parameterdaten speichern können, und diese über einen Programmbefehl je nach Erfordernis austauschen können.

HINWEISE

Die Zahl der PID-Anweisungen in einem Programm ist nicht beschränkt. Es ist jedoch darauf zu achten, daß sich die Datenbereiche (D3+) nicht überschneiden, um einen Datenkonflikt innerhalb der Steuerung zu vermeiden.

Die PID-Anweisung kann in einem Interrupt, in einem Unterprogramm, in einem STL-Programm oder in Programmsprüngen eingesetzt werden. Dabei ist der PID-Anweisung eine MOVP-Anweisung voranzustellen. In dieser MOVP-Anweisung sollte K0 auf ((S3+)+7) geschrieben werden, um möglichen Programmfehlern vorzubeugen.

**Abb. 7-45:**

Programmierbeispiel zur PID-Anweisung mit vorangestellter MOVP-Anweisung

C000330C

Die Sampling-Zeit (T_S) sollte länger als die Programm-Zykluszeit gewählt sein, da es andernfalls zu Fehlern kommt. Ist dies nicht der Fall, wird automatisch die Sampling-Zeit gleich der Zykluszeit gesetzt. Bei Anwendung der Interrupt-Anweisungen I6XX bis I8XX sollte die Sampling-Zeit nicht kürzer als die Interrupt-Zykluszeit sein.

Die Sampling-Zeit (T_S) kann aufgrund des Programmescans variieren. Der maximale Bereich der Abweichung liegt bei ($T_S - (\text{Programm-Zykluszeit})$) bis ($T_S + (\text{Programm-Zykluszeit})$). Diese Abweichung kann durch Einsatz der PID-Anweisung innerhalb einer getakteten Interrupt-Routine minimiert werden.

Die PID-Anweisung erlaubt die Anzeige von Alarmmeldungen bei Störungen im Prozeßablauf. Diese Alarmmeldungen können vom Anwender aktiviert und deaktiviert werden. Ebenso kann über benutzerdefinierte Parameter eingestellt werden, bei welchem Zustand eine Alarmmeldung erfolgen soll.

Die PID-Anweisung beinhaltet gewisse Fehlermeldungen, die helfen, ein auftretendes Problem zu lösen. Diese Meldungen werden im Datenregister D8067 gespeichert. Tritt ein Fehler auf, wird er durch Setzen von Sondermerker M8067 angezeigt. Tabelle 7-12 enthält eine Übersicht der Fehlermeldungen und ihrer Bedeutung.

Da die Steuerungen der FX1S-Serie keine Analogwerte einlesen können, müssen der SPS die Daten für die Regelung über eine serielle Verbindung zugeführt werden.

Fehlermeldung	Beschreibung	Auswirkung auf PID-Anweisung
K6705	Die PID-Anweisung verweist nicht auf Datenregister.	Die Ausführung stoppt.
K6706	Die angegebenen Datenregister liegen außerhalb des zulässigen Bereichs.	
K6730	Die Sampling-Zeit (T_s) liegt außerhalb des zulässigen Bereichs ($T_s < 0$).	
K6732	Der Einstellwert für den Input-Filter (α) liegt außerhalb des zulässigen Bereichs ($\alpha < 0$ oder $\alpha > 100$).	
K6733	Der Proportionalfaktor (K_p) liegt außerhalb des zulässigen Bereichs ($K_p < 0$).	
K6734	Die Integrierzeitkonstante (T_i) liegt außerhalb des zulässigen Bereichs ($T_i < 0$).	
K6735	Der Differentialfaktor (K_d) liegt außerhalb des zulässigen Bereichs ($K_d < 0$ oder $K_d \geq 101$).	
K6736	Die differentielle Zeitkonstante (T_D) liegt außerhalb des zulässigen Bereichs ($T_D < 0$).	Die Sampling-Zeit wird gleich der Zykluszeit gesetzt, und die Ausführung wird fortgesetzt.
K6740	Sampling-Zeit (T_s) \leq Programm-Zykluszeit	
K6742	Die Istwertveränderung liegt außerhalb des zulässigen Bereichs (Δ Istwert < -32768 oder Δ Istwert $> +32767$).	Die betroffenen Daten werden auf den erlaubten Grenzwert herauf-, bzw. herabgesetzt, und die Ausführung wird fortgesetzt.
K6743	Die Abweichung liegt außerhalb des zulässigen Bereichs ($\epsilon < -32768$ oder $\epsilon > +32767$).	
K6744	Das Integrierergebnis liegt außerhalb des zulässigen Bereichs ($-32768 - +32767$).	
K6745	Der Differentialfaktor (K_D) liegt über oder der Differenzialwert liegt außerhalb des zulässigen Bereichs.	
K6746	Das Differenzierergebnis liegt außerhalb des zulässigen Bereichs ($-32768 - +32767$).	
K6747	Das PID-Gesamtergebnis liegt außerhalb des zulässigen Bereichs ($-32768 - +32767$).	
K6750	Sollwert - Istwert < 150 oder der Regelkreis ist instabil (Die Differenz zwischen Soll- und Istwert schwankt zu stark und zu schnell)	Die Ausführung wird beendet.
K6751	Der Soll-Wert ist erheblich zu groß	
K6752	Große Schwankungen bei der Auto-Tuning-Funktion	

Tab. 7-12: Übersicht der in Register D8067 gespeicherten Fehlermeldungen

Beispiel ▾ Einsatz der PID-Anweisung

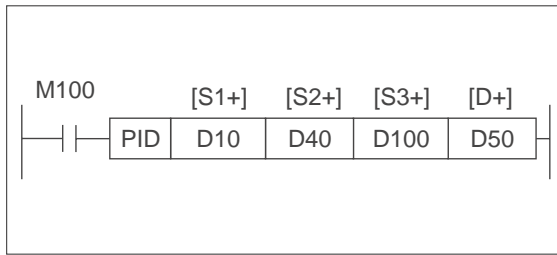


Abb. 7-46:
 Programmierbeispiel zur PID-Anweisung

C000331C

Die PID-Anweisung wird ausgeführt, sobald M100 gesetzt wird. Der Sollwert ist in D10 gespeichert, der Istwert wird nach D40 eingelesen, und der Ausgangswert wird nach D50 geschrieben. Die Regelparameter sind in den Datenregistern D100 bis D124 gespeichert.

Sollen analoge Werte geregelt werden, müssen dem SPS-Grundgerät zwei zusätzliche Sondermodule zur Handhabung der von der Steuerung eingelesenen und an die Steuerung auszugebenden Signale hinzugefügt werden.

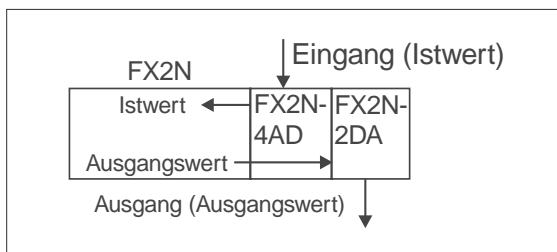


Abb. 7-47:
 Beispiel für ein Analogsystem

C000332C

Bei dem ersten Sondermodul handelt es sich um einen A/D-Wandler vom Typ FX2N-4AD, der die analogen Prozessdaten digitalisiert. Das Grundgerät verwendet eine FROM-Anweisung, um diese Daten in D40 abzulegen. Ein Beispiel für einen analogen Istwert wäre eine wechselnde Temperatur und damit ein wechselndes Spannungssignal.

Bei dem zweiten Sondermodul handelt es sich um einen D/A-Wandler vom Typ FX2N-2DA, der die in D50 gespeicherten digitalen Ausgangswerte analog ausgibt. Ein Beispiel für eine analoge Regelung wäre ein Stellventil, dessen Stellwert abhängig von einem Spannungspegel ist.

Die Regelparameter hängen von den spezifischen Gegebenheiten des Regelsystems ab.

Eine andere Möglichkeit für einen Regelausgangswert ist zum Beispiel die PWM-Anweisung (PWM, FNC 58), die fortlaufende Impulse ausgibt. Deren Pulsweite kann über den Ausgangswert bestimmt werden.

△

Einstellung der PID-Anweisung

Regelmethode	Auswahl über Datenregister			Beschreibung
	(S3+)+3(K _P)	(S3+)+4(T _I)	(S3+)+6(T _D)	
P	Anwenderwert	auf Null gesetzt	auf Null gesetzt	proportionale Regelung
PI	Anwenderwert	Anwenderwert	auf Null gesetzt	proportionale und integrale Regelung
PD	Anwenderwert	auf Null gesetzt	Anwenderwert	proportionale und differenzielle Regelung
PID	Anwenderwert	Anwenderwert	Anwenderwert	volle PID-Regelung

Tab. 7-13: Einstellung der PID-Anweisung

Vorwärts- und Rückwärts-Operationen ((S3+)+1, b0)

Die Bezeichnung der Vorwärts- und Rückwärts-Operation erscheint vom Namen her verwirrend. Von der Vorstellung her kommt es der Bewegung eines Punktes in einem Koordinatensystem am nächsten. Für die beiden Parameter kann man sich die folgende Situation vorstellen:

- Der Istwert (CV, Datenregister (S2+)) ist größer als der Sollwert (SP oder Datenregister (S1+)).
- Der Istwert (CV, Datenregister (S2+)) ist kleiner als der Sollwert (SP oder Datenregister (S1+)).

Die folgende Abb. stellt ein Koordinatensystem dar, in dem die Koordinatenachsen für den Sollwert, bzw. den Ausgangswert des Regelkreises, und den Istwert stehen. In der Abb. 7-47 stellt die gestrichelte Linie den Verlauf einer Vorwärtsbewegung und die durchgezogene Linie den Verlauf einer Rückwärtsbewegung dar.

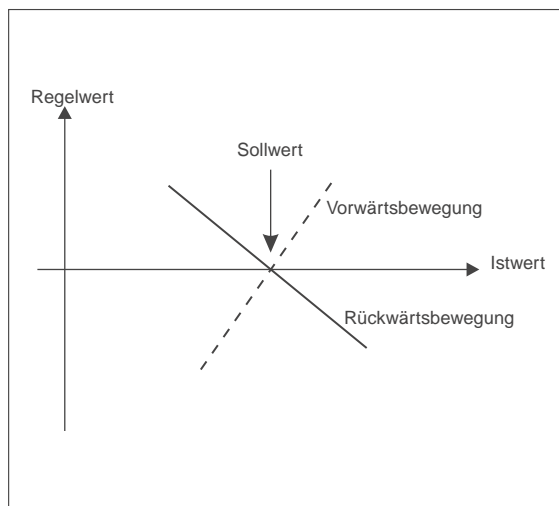


Abb. 7-48:

Darstellung der Vorwärts-/Rückwärtsbewegung in einem Koordinatensystem

C000343C

Die Vorwärtsbewegung besteht bei zu großem Istwert im runter und zurück regeln des Istwertes, bzw. bei zu kleinem Istwert im rauf und vorwärts regeln des Istwertes.

Die Rückwärtsbewegung liegt vor, wenn bei zu großem Istwert der Istwert runter und vorwärts geregelt wird, bzw. bei zu kleinem Istwert rauf und zurück geregelt wird.

In der Grafik ist kein Korrekturfaktor P, I oder D, oder eine Kombination aus diesen, berücksichtigt worden.

7.4 Anweisung mit Gleitkommazahlen (FX2N)

Übersicht der Anweisungen FNC 110 bis 132

Symbol	FNC	Bedeutung	Abschnitt
DECMP	110	Vergleich von Gleitkommazahlen	7.4.1
DEZCP	111	Vergleich von Gleitkommazahlen mit einem Bereich	7.4.2
DEBCD	118	Umwandlung des Gleitkommaformats ins wissenschaftliche Zahlenformat	7.4.3
DEBIN	119	Umwandlung des wissenschaftlichen Zahlenformats ins Gleitkommaformat	7.4.4
DEADD	120	Addition von Gleitkommazahlen	7.4.5
DESUB	121	Subtraktion von Gleitkommazahlen	7.4.6
DEMUL	122	Multiplikation von Gleitkommazahlen	7.4.7
DEDIV	123	Division von Gleitkommazahlen	7.4.8
DESQR	127	Quadratwurzeln von Gleitkommazahlen	7.4.9
INT	129	Umwandlung des Gleitkommaformats ins Dezimal-Format	7.4.10
SIN	130	Sinusberechnung mit Gleitkommazahlen	7.4.11
COS	131	Cosinusberechnung mit Gleitkommazahlen	7.4.12
TAN	132	Tangensberechnung mit Gleitkommazahlen	7.4.13

Tab. 7-14: Übersicht der Anweisung FNC 110 bis 132

7.4.1 Vergleich von Gleitkommazahlen (DECMP)

				DECMP		FNC 110		
				Vergleich von Gleitkommazahlen				CPU
							●	
Operanden	S1+	S2+	D+	Puls-Anweisung (P) ●	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))		Y, M, S, Es werden drei aufeinanderfolgende Adressen des Operanden verwendet		16 Bit	32 Bit	DECMP	13
					●	DECMP	13	

Funktion

Vergleichen von 2 Gleitkommazahlen mit Ausgabe der Vergleichsergebnisse

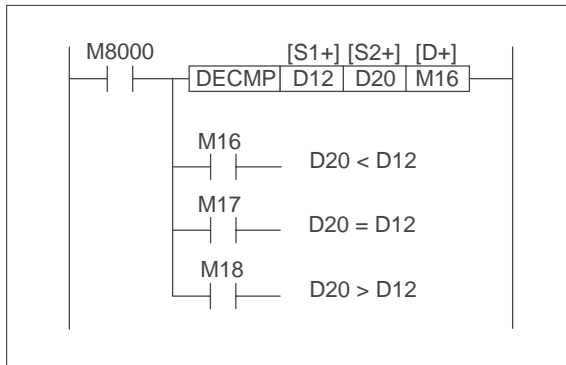
Beschreibung

- Die DECMP-Anweisung vergleicht die Gleitkommazahl ab (S1+) mit der Gleitkommazahl ab (S2+).
- Die Vergleichsergebnisse werden in jeweils 3 aufeinanderfolgenden Operanden gespeichert.
- Ist die Zahl ab (S2+) kleiner als die Zahl ab (S1+), wird der Bit-Operand (D+) gesetzt.
- Ist die Zahl ab (S2+) gleich der Zahl ab (S1+), wird der Bit-Operand ((D+)+1) gesetzt.
- Ist die Zahl ab (S2+) größer als die Zahl ab (S1+), wird der Bit-Operand ((D+)+2) gesetzt.

HINWEISE

Die angesprochenen Ausgangsoperanden bleiben nach Abschalten der Ausführungsbedingung der DECMP-Anweisung gesetzt.

Die Vergleiche werden algebraisch durchgeführt (z.B wird $-1,79 \times 10^{27}$ kleiner als $9,43 \times 10^{-15}$ erkannt).

Beispiel ▾**Abb. 7-49**

Programmierbeispiel zur DECMP-Anweisung

C000350C

Mit Setzen des Merkers M8000 wird die ab D20 (S2+) angegebene Gleitkommazahl mit der ab D12 (S1+) angegebenen Gleitkommazahl verglichen.

Ist die Zahl ab D20 kleiner als die Zahl ab D12, wird der Merker M16 gesetzt.

Ist die Zahl ab D20 gleich der Zahl ab D12, wird der Merker M17 gesetzt.

Ist die Zahl ab D20 größer als die Zahl ab D12, wird der Merker M18 gesetzt.

△

7.4.2 Vergleich von Gleitkommazahlen mit einem Bereich (DEZCP)

					DEZCP		FNC 111		
					Vergleich von Gleitkommazahlen mit einem Bereich				
					CPU	FX1S	FX1N	FX2N	
								●	
Operanden	S1+	S2+	S3+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))			Y, M, S, Es werden drei aufeinanderfolgende Adressen des Operanden verwendet		●	16 Bit	32 Bit	DEZCP
							●	DEZCPP	13

Funktion

Vergleichen einer Gleitkommazahl mit einem Bereich mit Ausgabe der Vergleichsergebnisse

Beschreibung

- Die DEZCP-Anweisung vergleicht die Gleitkommazahl ab (S3+) mit dem Bereich zwischen (S1+) und (S2+).
- Die Vergleichsergebnisse werden in jeweils 3 aufeinanderfolgenden Operanden gespeichert.
- Ist die Zahl ab (S3+) kleiner als die Zahlen zwischen (S1+) und (S2+), wird der Bit-Operand (D+) gesetzt.
- Ist die Zahl ab (S3+) gleich einer Zahl zwischen (S1+) und (S2+), wird der Bit-Operand ((D+)+1) gesetzt.
- Ist die Zahl ab (S3+) größer als die Zahlen zwischen (S1+) und (S2+), wird der Bit-Operand ((D+)+2) gesetzt.

HINWEISE

Die angesprochenen Ausgangsoperanden bleiben nach Abschalten der Ausführungsbedingung der DEZCP-Anweisung gesetzt.

Die Vergleiche werden algebraisch durchgeführt (z.B wird $-1,79 \times 10^{27}$ kleiner als $9,43 \times 10^{-15}$ erkannt).

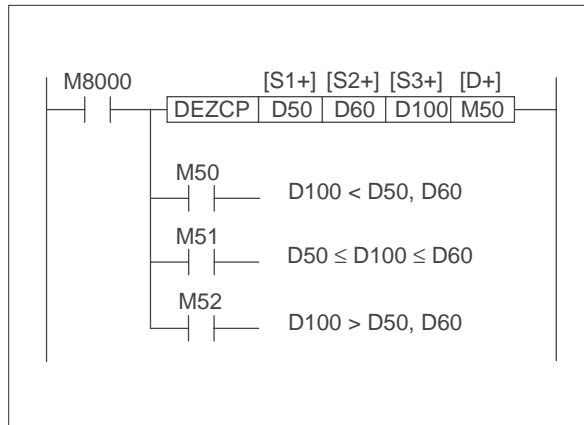
Beispiel ▾

Abb. 7-50:
 Programmierbeispiel zur DEZCP-
 Anweisung

C000351C

Mit Setzen des Merkers M8000 wird die ab D100 (S3+) angegebene Gleitkommazahl mit den Zahlen in dem Bereich zwischen D50 (S1+) und D60 (S2+) verglichen.

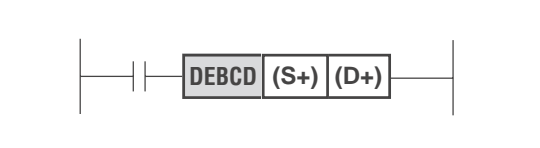
Ist die Zahl ab D100 kleiner als die Zahlen zwischen D50 und D60, wird der Merker M50 gesetzt.

Ist die Zahl ab D100 gleich einer Zahl zwischen D50 und D60, wird der Merker M51 gesetzt.

Ist die Zahl ab D100 größer als die Zahlen zwischen D50 und D60, wird der Merker M52 gesetzt.

△

7.4.3 Umwandlung des Gleitkommaformats ins wissenschaftliche Zahlenformat (DEBCD)

				DEBCD		FNC 118	
				Umwandlung vom Gleitkommaformat in das wissenschaftliche Zahlenformat			
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D (Gleitkommazahl (32 Bits))	D Es werden 2 aufeinanderfolgende Adressen der Operanden verwendet		16 Bit	32 Bit	DEBCD	9
						DEBCDP	9

Funktion

Umwandeln einer Zahl im Gleitkommaformat in eine Zahl im wissenschaftlichen Zahlenformat.

Beschreibung

- Die im Gleitkommaformat angegebene Zahl ab (S+) wird in das wissenschaftliche Zahlenformat konvertiert und ab (D+) gespeichert.
- Die Mantisse wird in (D+) gespeichert.
- Der Exponent wird in ((D+)+1) gespeichert.

HINWEISE

Um das Konvertierungsergebnis mit maximaler Genauigkeit darzustellen, wird die Mantisse (D+) mit 0 oder einem Wert zwischen 1000 und 9999 angegeben. Die Angabe des Exponenten ((D+)+1) wird dementsprechend korrigiert (z.B. wird $3,4567 \times 10^{-5}$ (S+, (S+)+1) konvertiert und als 3456 (D+) und -8 ((D+)+1) gespeichert).

Beispiel ▾

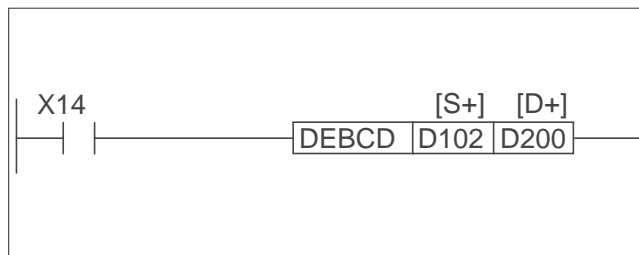


Abb. 7.51: Programmierbeispiel zur DEBCD-Anweisung

C000352C

Mit Setzen des Eingangs X14 wird die in D102 und D103 angegebene Zahl im Gleitkommaformat in das wissenschaftliche Zahlenformat konvertiert und anschließend ab D200 gespeichert.

Die Mantisse wird in D200 gespeichert.

Der Exponent wird in D201 gespeichert.

△

7.4.4 Umwandlung wissenschaftliches Zahlenformat ins Gleitkommaformat (DEBIN)

		DEBIN		FNC 119			
		Umwandlung vom wissenschaftlichen Zahlenformat in das Gleitkommaformat					
		CPU	FX1S	FX1N	FX2N		
					●		
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D Es werden 2 aufeinanderfolgende Adressen des Operanden verwendet	D (Gleitkommazahl (32 Bits))	●	16 Bit	32 Bit	DEBIN	9
					●	DEBINP	9

Funktion

Umwandeln einer Zahl im wissenschaftlichen Zahlenformat in eine Zahl im Gleitkommaformat.

Beschreibung

- Die im wissenschaftlichen Zahlenformat angegebene Zahl ab (S+) wird in das Gleitkommaformat konvertiert und ab (D+) gespeichert.
- Die Mantisse wird in (S+) angegeben.
- Der Exponent wird in ((S+)+1) angegeben.

HINWEISE

Um das Konvertierungsergebnis mit maximaler Genauigkeit darzustellen, muß die Mantisse (S+) mit 0 oder einem Wert zwischen 1000 und 9999 angegeben werden. Die Angabe des Exponenten ((S+)+1) muß dementsprechend korrigiert werden (z.B. werden die für Mantisse und Exponent angegebenen Werte 5432 (S+) und 12 ((S+)+1) in die Zahl $5,432 \times 10^9$ (D+), (D+)+1 im Gleitkommaformat konvertiert).

Beispiel ▾

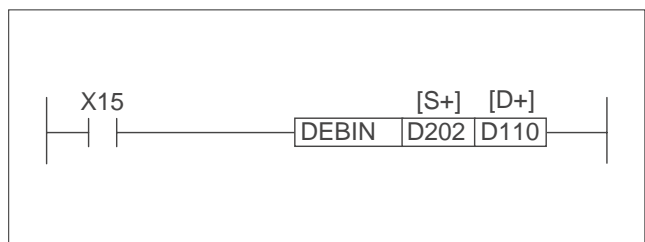


Abb. 7-52:
Programmierbeispiel zur DEBIN-Anweisung

C000353C

Mit Setzen des Eingangs X15 wird die in D202 und D203 angegebene Zahl im wissenschaftlichen Zahlenformat in das Gleitkommaformat konvertiert und anschließend ab D110 gespeichert.

Die Mantisse wird in D202 angegeben.

Der Exponent wird in D203 angegeben.

7.4.5 Addition von Gleitkommazahlen (DEADD)

				DEADD		FNC 120		
				Addition von Gleitkommazahlen				CPU
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))				D (Gleitkommazahl (32 Bits))	16 Bit	32 Bit	DEADD
				●		●	DEADDP	13

Funktion

Addieren von zwei Gleitkommazahlen und speichern des Ergebnisses

Beschreibung

- Die ab (S1+) angegebene Gleitkommazahl wird mit der Gleitkommazahl ab (S2+) addiert. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Eingegebene Konstanten (K, H) werden vor der Addition automatisch in Gleitkommazahlen umgewandelt.
- Als Quelle und Ziel kann derselbe Operand verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus.
- Ist das Additionsergebnis 0, wird das Zero-Flag M8020 gesetzt.
- Ist das Additionsergebnis größer als der maximal zulässige Wert, wird das Carry-Flag M8022 gesetzt.
- Ist das Additionsergebnis kleiner als der minimal zulässige Wert, wird das Borrow-Flag M8021 gesetzt.

HINWEISE

Die Additionen werden nach den geltenden mathematischen Gesetzmäßigkeiten ausgeführt (z.B. liefert die Addition von $2,3456 \times 10^2 + (-5,6 \times 10^{-1})$ das Ergebnis $2,34 \times 10^2$).

Beispiel

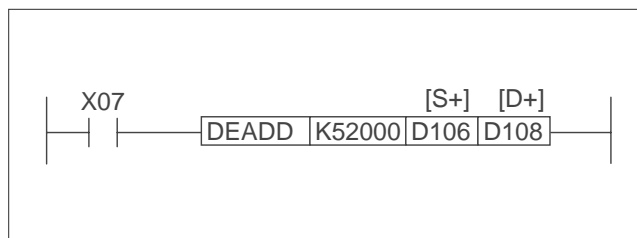


Abb. 7-53:
 Programmierbeispiel zur DEADD-Anweisung

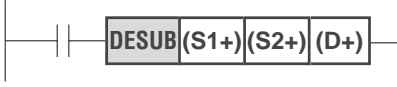
C000354C

Mit Setzen des Eingangs X7 wird zu der Konstanten K52000 die ab D106 angegebene Gleitkommazahl addiert.

Das Ergebnis wird ab D108 gespeichert.



7.4.6 Subtraktion von Gleitkommazahlen (DESUB)

				DESUB		FNC 121		
				Subtraktion von Gleitkommazahlen				
CPU				FX1S	FX1N	FX2N		
						●		
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H (Integer-Werte werden automatisch in Gleitkommazahlen konvertiert) D (Gleitkommazahl (32 Bits))				D (Gleitkommazahl (32 Bits))	●	16 Bit	32 Bit
						●	DESUBP	13

Funktion

Subtrahieren von zwei Gleitkommazahlen und speichern des Ergebnisses

Beschreibung

- Die ab (S2+) angegebene Gleitkommazahl wird von der Gleitkommazahl ab (S1+) subtrahiert. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Eingegebene Konstanten (K, H) werden vor der Subtraktion automatisch in Gleitkommazahlen umgewandelt.
- Als Quelle und Ziel kann derselbe Operand verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus.
- Ist das Subtraktionsergebnis 0, wird das Zero-Flag M8020 gesetzt.
- Ist das Subtraktionsergebnis größer als der maximal zulässige Wert, wird das Carry-Flag M8022 gesetzt.
- Ist das Subtraktionsergebnis kleiner als der minimal zulässige Wert, wird das Borrow-Flag M8021 gesetzt.

HINWEISE

Die Subtraktionen werden nach den geltenden mathematischen Gesetzmäßigkeiten ausgeführt (z.B. liefert die Subtraktion von $2,3456 \times 10^2 - 5,6 \times 10^{-1}$ das Ergebnis $2,34 \times 10^2$).

Beispiel ▾

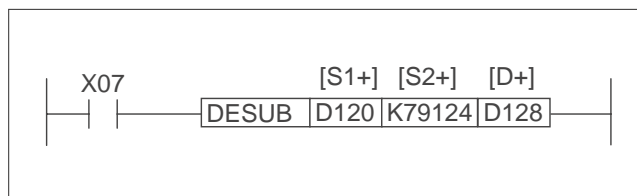


Abb. 7-54:
Programmierbeispiel zur DESUB-Anweisung

C000355C

Mit Setzen des Eingangs X17 wird von der Gleitkommazahl ab D120 die Konstante K79124 subtrahiert.

Das Ergebnis wird ab D128 gespeichert.

△

7.4.7 Multiplikation von Gleitkommazahlen (DEMUL)

				DEMUL		FNC 122		
				Multiplikation von Gleitkommazahlen				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))				D (Gleitkommazahl (32 Bits))	●	16 Bit	32 Bit
						●	DEMULP	13

Funktion

Multiplizieren von zwei Gleitkommazahlen und speichern des Ergebnisses

Beschreibung

- Die ab (S1+) angegebene Gleitkommazahl wird mit der Gleitkommazahl ab (S2+) multipliziert. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Eingegebene Konstanten (K, H) werden vor der Multiplikation automatisch in Gleitkommazahlen umgewandelt.
- Als Quelle und Ziel kann derselbe Operand verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus.

HINWEIS Die Multiplikationen werden nach den geltenden mathematischen Gesetzmäßigkeiten ausgeführt.

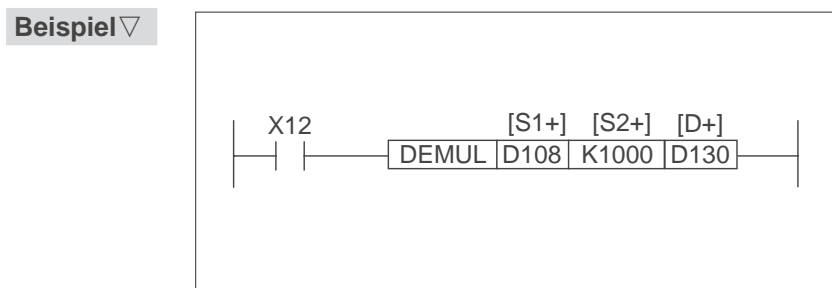


Abb. 7-55:
Programmierbeispiel zur DEMUL-Anweisung

C000356C

Mit Setzen des Merkers M12 wird die Gleitkommazahl ab D108 mit der Konstanten K1000 multipliziert.

Das Ergebnis wird ab D130 gespeichert.



7.4.8 Division von Gleitkommazahlen (DEDIV)

				DEDIV		FNC 123		
				Division von Gleitkommazahlen				
				CPU	FX1S	FX1N	FX2N	
							●	
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))				●	16 Bit	32 Bit	DEDIV
D (Gleitkommazahl (32 Bits))				●			DEDIVP	13

Funktion

Dividieren von zwei Gleitkommazahlen und speichern des Ergebnisses

Beschreibung

- Die ab (S1+) angegebene Gleitkommazahl wird durch die Gleitkommazahl ab (S2+) dividiert. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Eingegebene Konstanten (K, H) werden vor der Division automatisch in Gleitkommazahlen umgewandelt.
- Als Quelle und Ziel kann derselbe Operand verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus.

HINWEIS

Die Divisionen werden nach den geltenden mathematischen Gesetzmäßigkeiten ausgeführt.

Fehlerquelle

Wenn der Wert ab (S2+) mit 0 angegeben wird, erfolgt die Meldung des Fehlers „Division durch 0“ und die Verarbeitung wird abgebrochen.

Beispiel ▾

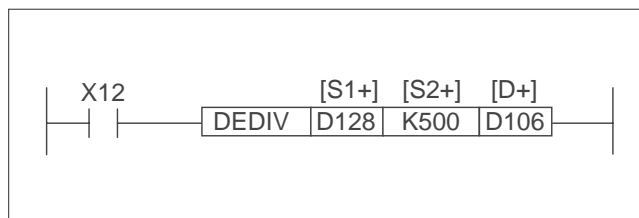


Abb. 7-56:
 Programmierbeispiel zur DEDIV-Anweisung

C000357C

Mit Setzen des Eingangs X10 wird die Gleitkommazahl ab D128 durch die Konstante K500 dividiert.

Das Ergebnis wird ab D106 gespeichert.



7.4.9 Quadratwurzel aus Gleitkommazahlen (DESQR)

		DESQR		FNC 127			
		Quadratwurzel aus Gleitkommazahlen					
		CPU	FX1S	FX1N	FX2N		
					●		
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H Integer-Werte werden automatisch in Gleitkommazahlen konvertiert D (Gleitkommazahl (32 Bits))	D (Gleitkommazahl (32 Bits))	●	16 Bit	32 Bit	DESQR	9
					●	DESQRP	9

Funktion

Berechnen der Quadratwurzel aus einer Gleitkommazahl und speichern des Ergebnisses

Beschreibung

- Es wird aus der ab (S+) angegebenen Gleitkommazahl die Quadratwurzel berechnet. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Eingegebene Konstanten (K, H) werden vor der Wurzelberechnung automatisch in Gleitkommazahlen umgewandelt.
- Als Quelle und Ziel kann derselbe Operand verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus.
- Wenn das Ergebnis der Wurzelberechnung 0 beträgt, wird das Zero-Flag M8020 gesetzt.

HINWEIS

Die Wurzelberechnungen werden nach den geltenden mathematischen Gesetzmäßigkeiten ausgeführt.

Fehlerquelle

Wenn ab (S+) ein negativer Wert eingegeben wird, erfolgt eine Fehlermeldung und das Error-Flag M8067 wird gesetzt.

Beispiel ▾



Abb. 7-57:
Programmierbeispiel zur DESQR-Anweisung


C000358C

Mit Setzen des Merkers M24 wird aus der Gleitkommazahl ab D302 die Quadratwurzel berechnet.

Das Ergebnis wird ab D510 gespeichert.



7.4.10 Umwandlung des Gleitkommaformats ins Dezimal-Format (INT)

		INT		FNC 129			
		Umwandlung des Gleitkommaformats ins Dezimal - Format					
CPU		FX1S	FX1N	FX2N			
				●			
Operanden	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	D (Gleitkommazahl (32 Bits))	D (Dezimal - Format) INT, INTP (16 Bits) DINT, DINTP (32 Bits)		●	16 Bit	32 Bit	INT, INTP
			●	●		DINT, DINTP	9

Funktion

Konvertierung einer Gleitkommazahl in das Dezimal-Format

Beschreibung

- Die ab (S+) angegebene Gleitkommazahl wird auf den nächst kleineren Integer-Wert abgerundet und ab (D+) gespeichert.
- Der Quelloperand ist immer ein Doppelwortoperand.
- Bei Verwendung der INT-Anweisung ist der Zieloperand ein Wortoperand.
- Bei Verwendung der DINT-Anweisung ist der Ziel-Operand ein Doppelwortoperand.
- Die INT-Anweisung ist die Umkehrfunktion der FLT-Anweisung.
- Wenn das Ergebnis der Konvertierung 0 beträgt, wird das Zero-Flag M8020 gesetzt.
- Wird ab (S+) keine ganze Zahl angegeben, wird diese Zahl auf den nächst kleineren Integer-Wert abgerundet und das Borrow-Flag M8021 gesetzt.
- Wenn der konvertierte Integer-Wert außerhalb des Speicherbereichs des Zieloperanden liegt, tritt ein Überlauf auf, und das Carry-Flag M8022 wird gesetzt.

HINWEIS

Beim Auftreten eines Überlaufs ist das Ergebnis im Zieloperanden fehlerhaft.

Beispiel ▾

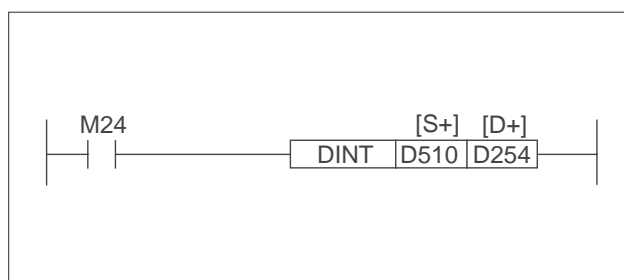


Abb. 7-58:
Programmierbeispiel zur DINT-
Anweisung

C000359C

Mit Setzen des Merkers M25 wird die Gleitkommazahl in D510 und D511 auf den nächst kleineren Integer-Wert abgerundet und das Borrow-Flag M8021 gesetzt.

Das Ergebnis wird in D254 und D255 gespeichert.

△

7.4.11 Sinusberechnung mit Gleitkommazahlen (DSIN)

		DSIN		FNC 130				
		Sinusberechnung mit Gleitkommazahlen						
Operanden		S+ D (Gleitkommazahl in Radiant (32 Bits))	D+ D (Gleitkommazahl (32 Bits))	Puls-Anweisung (P) ●	CPU	FX1S	FX1N	FX2N
								●
					Verarbeitung		Programmschritte	
					16 Bit	32 Bit	DSIN	9
						●	DSINP	9

Funktion

Berechnen des Sinus einer Gleitkommazahl und speichern des Ergebnisses

Beschreibung

- Es wird der Sinus aus der ab (S+) angegebenen Gleitkommazahl berechnet. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Die Werte im Quell- und Zieloperanden haben Gleitkommaformat.
- Der ab (S+) angegebene Winkelwert muß ein Wert zwischen 0 und 360° (0 und 2 π rad) sein. Die Angabe des Winkelwertes muß in Radiant erfolgen ($\text{Grad} \times \pi / 180$ [rad]).

Beispiel ▾

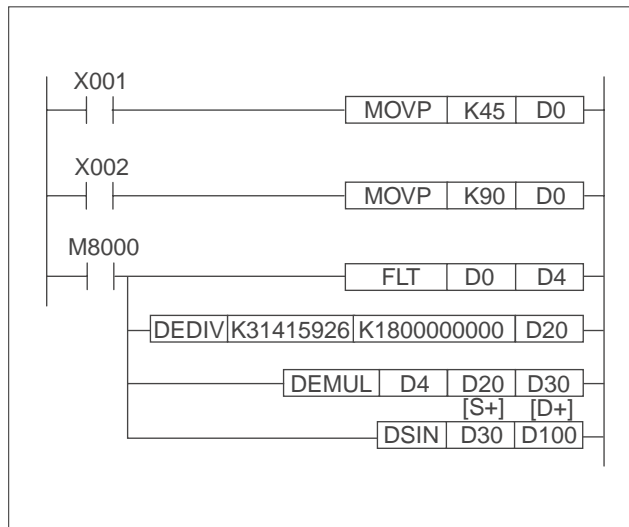


Abb. 7-59:
 Programmierbeispiel zur DSIN-
 Anweisung mit Radianten-
 umwandlung

C000360C

Mit positiver Flanke des Eingangs X1 wird die Konstante K45 (45°) in D0 geschrieben.

Mit positiver Flanke des Eingangs X2 wird die Konstante K90 (90°) in D0 geschrieben.

Mit Setzen des Merkers M8000 wird der Wert aus D0 in eine Gleitkommazahl konvertiert und in D4 und D5 gespeichert.

Mit den DEDIV- und DEMUL-Anweisungen erfolgt die Umrechnung dieses Wertes in Radiant.

Das Ergebnis wird in D30 und D31 gespeichert.

Mit der DSIN-Anweisung wird daraus der Sinus berechnet. Das Ergebnis wird in D100 und D101 gespeichert.

△

7.4.12 Cosinusberechnung mit Gleitkommazahlen (DCOS)

		DCOS		FNC 131				
		Cosinusberechnung mit Gleitkommazahlen						
Operanden		S+ (Gleitkommazahl in Radiant (32 Bits))	D+ (Gleitkommazahl (32 Bits))	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					16 Bit	32 Bit	DCOS	9
				●	●	DCOSP	9	

Funktion

Berechnen des Cosinus einer Gleitkommazahl und speichern des Ergebnisses

Beschreibung

- Es wird der Cosinus aus der ab (S+) angegebenen Gleitkommazahl berechnet. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Die Werte im Quell- und Zieloperanden haben Gleitkommaformat.
- Der ab (S+) angegebene Winkelwert muß ein Wert zwischen 0 und 360° (0 und 2 π rad) sein. Die Angabe des Winkelwertes muß in Radiant erfolgen (Grad x π / 180 [rad]).

Beispiel ▾

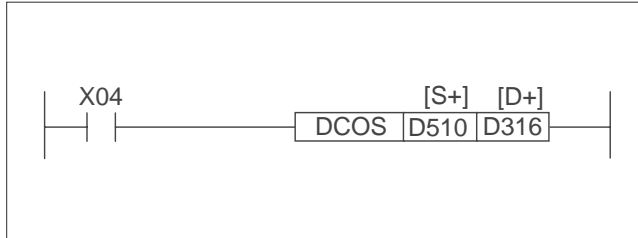


Abb. 7-60:
Programmierbeispiel zur DCOS-Anweisung

C000361C

Mit Setzen des Eingangs X4 wird der Cosinuswert der Winkelangabe in Radiant (D510, D511) berechnet (Umrechnung Grad in Radiant siehe 7.5.11).

Das Ergebnis wird in D316 und D317 gespeichert.

△

7.4.13 Tangensberechnung mit Gleitkommazahlen (DTAN)

		DTAN		FNC 132				
		Tangensberechnung mit Gleitkommazahlen						
Operanden		S+ (Gleitkommazahl in Radiant (32 Bits))	D+ (Gleitkommazahl (32 Bits))	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					16 Bit	32 Bit	DTAN	9
				●	●	DTANP	9	

Funktion

Berechnen des Tangens einer Gleitkommazahl und speichern des Ergebnisses

Beschreibung

- Es wird der Tangens aus der ab (S+) angegebenen Gleitkommazahl berechnet. Das Ergebnis wird ab (D+) gespeichert.
- Es werden für jeden Operanden jeweils 2 aufeinanderfolgende Register verwendet.
- Die Werte im Quell- und Zieloperanden haben Gleitkommaformat.
- Der ab (S+) angegebene Winkelwert muß ein Wert zwischen 0 und 360° (0 und 2 π rad) sein. Die Angabe des Winkelwertes muß in Radiant erfolgen (Grad x π / 180 [rad]).

Beispiel ▾

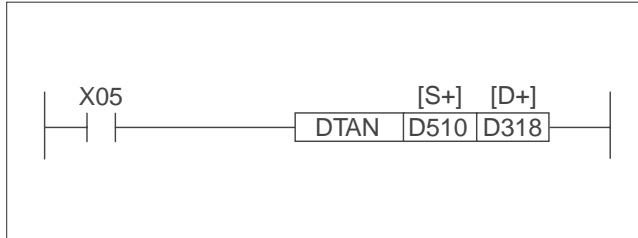


Abb. 7-61:
Programmierbeispiel zur DTAN-Anweisung

C000362C

Mit Setzen des Eingangs X5 wird der Tangenswert der Winkelangabe in Radiant (D510, D511) berechnet (Umrechnung Grad in Radiant siehe 7.5.11).

Das Ergebnis wird in D318 und D319 gespeichert.

△

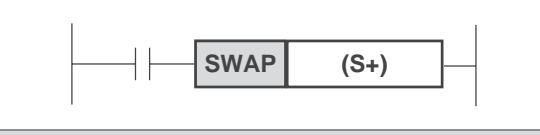
7.5 Datenverarbeitungsanweisungen

Übersicht der Anweisung FNC 147 (FX2N)

Symbol	FNC	Bedeutung	Abschnitt
SWAP	147	High - Low - Byte - Tausch	7.5.1

Tab. 7-15: Übersicht der Anweisung FNC 147

7.5.1 High-Low-Byte-Tausch (SWAP)

		SWAP		FNC 147		
		High-Low-Byte-Tausch				
Operanden	S+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
			CPU	FX1S		FX1N
	KnY, KnM, KnS, T, C, D, V, Z	●	16 Bit	32 Bit	SWAP, SWAPP	5
			●	●	DSWAP, DSWAPP	9

Funktion

Tauschen des High- und Low-Bytes eines Operanden

Beschreibung

- Bei Verwendung der SWAP-Anweisung werden das High- und Low-Byte des Operanden (D+) getauscht.
- Bei Verwendung der DSWAP-Anweisung werden jeweils die High- und Low-Bytes der Operanden (D+) und ((D+)+1) getauscht.
- Diese Operation wird in jedem Programmzyklus erneut ausgeführt. Um eine einmalige Ausführung zu gewährleisten, sind gepulste Anweisungen zu verwenden oder Verriegelungen einzusetzen.

Beispiel ▾

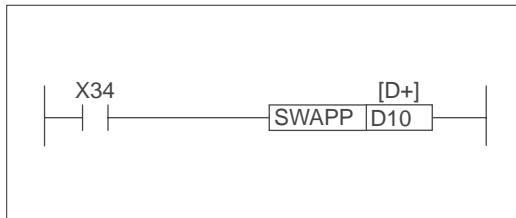


Abb. 7-62:
 Programmierbeispiel zur SWAPP-Anweisung

C000363C

Mit positiver Flanke des Eingangs X34 werden das High- und Low-Byte von D10 getauscht.

Operand	Byte	vor Ausführen der Anweisung	nach Ausführen der Anweisung
D10	1	1FH	8BH
	2	8BH	1FH

Tab. 7-21:
 Byte-Tausch bei der SWAPP-Anweisung

Wird statt der SWAPP-Anweisung die DSWAPP-Anweisung verwendet, werden mit positiver Flanke von X34 jeweils die High- und Low-Bytes in D10 und D11 getauscht.

Operand	Byte	vor Ausführen der Anweisung	nach Ausführen der Anweisung
D10	1	1FH	8BH
	2	8BH	1FH
D11	1	C4H	35H
	2	35H	C4H

Tab. 7-22:
 Byte-Tausch bei der DSWAPP-Anweisung



7.6 Positionieranweisungen (FX1S/FX1N)

Übersicht der Anweisungen FNC 155 bis 159

Symbol	FNC	Bedeutung	Abschnitt
ABS	155	Lesen der absoluten Ist-Position	7.6.1
ZRN	156	Referenzpunkt anfahren	7.6.2
PLSV	157	Ausgabe von Impulsen mit variabler Frequenz	7.6.3
DRVI	158	Positionieren auf einen Inkrementalwert	7.6.4
DRVA	159	Positionieren auf einen Absolutwert	7.6.5

Tab. 7-16: Übersicht der Anweisungen FNC 155 bis 159

7.6.1 Lesen der absoluten Ist-Position (DABS)

				DABS		FNC 155		
				Lesen der absoluten Ist-Position				
				CPU	FX1S	FX1N	FX2N	
					●	●		
Operanden	S+	D1+	D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	X, Y, M, S	Y, M, S	KnY, KnM, KnS, T, C, D, V, Z			16 Bit	32 Bit	DABS

Funktion

Mit dieser Anweisung kann die absolute Position eines angeschlossenen MITSUBISHI Servo-Verstärkers vom Typ MR-H oder MR-J2 erfasst werden. Der Servo-Antrieb muss in der Lage sein, die Position absolut zu erfassen.

Beschreibung

- Die ABS-Anweisung ist eine 32-Bit-Anweisung. Geben Sie aus diesem Grund die Anweisung immer als "DABS" ein.
- Das Lesen der Position beginnt, wenn die Startbedingung der DABS-Anweisung erfüllt ist. Wenn das Lesen abgeschlossen ist, wird M8029 gesetzt. Wenn die Startbedingung während des Lesens ausgeschaltet wird, wird der Lesevorgang unterbrochen.
- Bei der Projektierung des Antriebes ist zu beachten, dass der Servo-Verstärker vor oder gleichzeitig mit der SPS eingeschaltet wird.
- Die absolute Position wird in den Operanden abgelegt, der in D2+ angegeben wurde. Dies kann ein Wort-Operand sein. Jedoch sollte der Wert später im Programm in die Sonderregister D8141 und D8140 transferiert werden.
- Die Startbedingung der DABS-Anweisung sollte immer, auch nachdem die Position erfasst wurde, eingeschaltet sein. Wenn die Startbedingung nach dem Lesen der Position unwahr wird, wird das Signal "Servo-ON" (SON) rückgesetzt und der Antrieb gestoppt.
- Auch wenn der Servoantrieb die absolute Position erfassen kann, sollte nach dem Einschalten des Antriebes eine Referenzfahrt gemacht werden.

Beispiel ▾

ABS-Anweisung

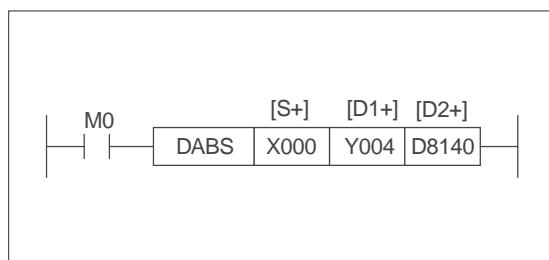


Abb. 7-63:
 Programmbeispiel zum Einsatz der ABS-Anweisung

C000400C



7.6.2 Referenzpunkt anfahren (ZRN)

					ZRN		FNC 156		
					Referenzpunkt anfahren				CPU
						●	●		
Operanden	S1+	S2+	S3+	D	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z,		X, Y, M, S	Nur Y0 oder Y1			16 Bit	32 Bit	ZRN
						●	●	DZRN	13

Funktion

Mit dieser Anweisung wird eine Fahrt zum Referenzpunkt (Nullpunkt) eingeleitet. Während der Positionierung wird von der SPS die Ist-Position gespeichert und abhängig von der Bewegungsrichtung erhöht oder erniedrigt. Dadurch ist die Maschinenposition in der SPS stets aktuell. Bei einem Spannungsausfall verliert die SPS diesen Positionswert. Deshalb wird nach dem Einschalten der Steuerung oder zum Einrichten der Maschine eine Referenzfahrt zu einer definierten Position gemacht.

Beschreibung

- In S1+ wird die Geschwindigkeit angegeben, mit der die Referenzfahrt gemacht werden soll. Bei Wort-Operanden sind 10 bis 32,767 Hz möglich, bei 32-Bit-Operanden geht der Bereich von 10 bis 100 kHz.
- Die Schleichgeschwindigkeit im Bereich von 10 bis 32,767 Hz wird in S2+ angegeben.
- S3+ enthält den Schalter zur Einleitung der Schleichfahrt in der Nähe des Referenzpunktes. Wenn in S3+ ein anderer Operand als ein Eingang (X) angegeben wird, wird der Zeitpunkt des Bremsens von der Zykluszeit der SPS beeinflusst und die Abweichung vom Referenzpunkt ist evtl. zu gross.
- Zur Ausgabe der Impulse können nur die Ausgänge Y0 und Y1 verwendet werden. Da die Impulse mit hoher Frequenz ausgegeben werden, müssen Steuerungen mit Transistorausgängen verwendet werden. Relaiskontakte nutzen sich in diesem Fall vorzeitig ab und sind daher ungeeignet.
Zur Erzeugung steiflankiger Ausgangssignale sollte der Laststrom der Transistorausgänge mindestens 200 mA betragen. Eventuell ist die Verwendung von Pull-up-Widerständen notwendig.
- Wenn der Sondermerker M8014 gesetzt ist, wird dem Servo-Verstärker signalisiert, wenn der Referenzpunkt erreicht wird.
- Die folgenden Operanden stehen für Positionieraufgaben zur Verfügung:

Operand	Beschreibung
D8141 (höherwertiges Wort) D8140 (niederwertiges Wort)	Istwert von Y0 (32 Bit)
D8143 (höherwertiges Wort) D8142 (niederwertiges Wort)	Istwert von Y1 (32 Bit)
M8145	Impulsausgabe an Y0 sofort stoppen
M8146	Impulsausgabe an Y1 sofort stoppen
M8147	Überwachung der Impulsausgabe an Y0 (BUS/READY)
M8148	Überwachung der Impulsausgabe an Y1 (BUS/READY)

Tab. 7-17: Operanden zur Positionierung

- Bei einem MITSUBISHI Servo-Verstärkers vom Typ MR-H oder MR-J2, der in der Lage ist, die absolute Position zu erfassen, kann die Ist-Position mit der ABS-Anweisung (FNC155) gelesen werden.

HINWEISE

Der Referenzpunkt kann nicht aus jeder Richtung angefahren werden. Bei der Referenzfahrt wird immer rückwärts in Richtung des Referenzpunkts gefahren.

Beachten Sie bei der Programmierung der Startbedingung die zeitlichen Abläufe.

Beispiel ▾

ZRN-Anweisung

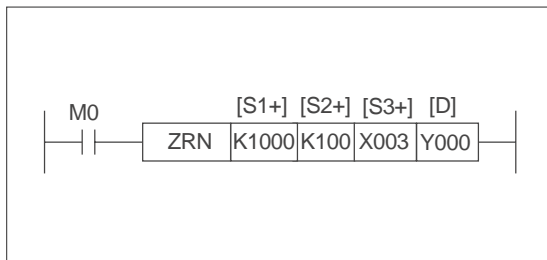
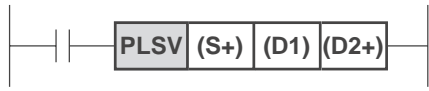


Abb. 7-64:
Programmbeispiel zum Einsatz der ZRN-Anweisung

C000401C



7.6.3 Ausgabe von Impulsen mit variabler Frequenz (PLSV)

				PLSV		FNC 157	
				Impulsausgabe			
CPU				FX1S	FX1N	FX2N	
				●	●		
Operanden	S+	D1	D2+	Puls-Anweisung (P)	Verarbeitung	Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	Nur Y0 oder Y1	Y, M, S		16 Bit ●	32 Bit ●	PLSV
						DPLSV	17

Funktion

Ausgabe von Impulsen mit variabler Frequenz an Y0 oder Y1 und eines Signals zur Festlegung der Drehrichtung

Beschreibung

- In S1+ wird die Frequenz der Ausgangsimpulse angegeben. Bei Wort-Operanden umfasst der Bereich 10 bis 32,767 Hz, bei 32-Bit-Operanden ist die Ausgabe einer Frequenz von 10 bis 100 kHz möglich.
- Zur Ausgabe der Impulse können nur die Ausgänge Y0 und Y1 verwendet werden. Da die Impulse mit hoher Frequenz ausgegeben werden, müssen Steuerungen mit Transistorausgängen verwendet werden. Relaiskontakte nutzen sich in diesem Fall vorzeitig ab und sind daher ungeeignet.
Zur Erzeugung steiflankiger Ausgangssignale sollte der Laststrom der Transistorausgänge mindestens 200 mA betragen. Eventuell ist die Verwendung von Pull-up-Widerständen notwendig.
- In D2+ wird ein Operand angegeben, der dem Servoverstärker die Drehrichtung vorgibt. Wenn der Operand nicht gesetzt ist, ist Linkslauf angewählt, bei gesetztem Operanden Rechtslauf.
- Die Drehrichtung wird durch das Vorzeichen der in S+ angegebenen Frequenz bestimmt. Ein positiver Wert bedeutet Rechtslauf, bei einem negativen Wert wird der Antrieb im Linkslauf betrieben.
- Während der Ausgabe von Impulsen kann die Frequenz der Impulse (in S+) geändert werden.
- Beim Beginn und beim Beenden der Impulsausgabe wird keine Beschleunigungs- oder Verzögerungsrampe durchlaufen. Wenn dies gefordert wird, muss mit der RAMP-Anweisung (FNC67) die Ausgangsfrequenz verändert werden.
- Wenn die Startbedingung der PLSV-Anweisung während der Impulsausgabe nicht mehr erfüllt ist, wird der Antrieb angehalten, ohne dass eine Verzögerungsrampe durchlaufen wird.
- Nach dem Rücksetzen der Startbedingung kann die PLSV-Anweisung nicht erneut gestartet werden, solange die Sondermerker M8147 (Überwachung von Y0) und M8148 (Überwachung von Y0) gesetzt sind.

- Die folgenden Operanden stehen für Positionieraufgaben zur Verfügung:

Operand	Beschreibung
D8141 (höherwertiges Wort) D8140 (niederwertiges Wort)	Istwert von Y0 (32 Bit)
D8143 (höherwertiges Wort) D8142 (niederwertiges Wort)	Istwert von Y1 (32 Bit)
M8145	Impulsausgabe an Y0 sofort stoppen
M8146	Impulsausgabe an Y1 sofort stoppen
M8147	Überwachung der Impulsausgabe an Y0 (BUS/READY)
M8148	Überwachung der Impulsausgabe an Y1 (BUS/READY)

Tab. 7-18: Operanden zur Positionierung

HINWEIS

Beachten Sie bei der Programmierung der Startbedingung die zeitlichen Abläufe.

Beispiel ▾

PLSV-Anweisung

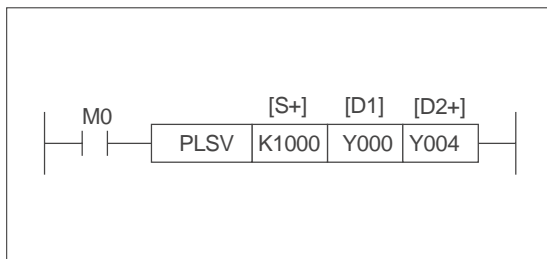


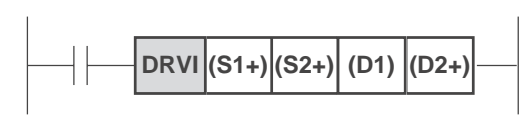
Abb. 7-65:

Programmbeispiel zum Einsatz der PLSV-Anweisung

C000402C



7.6.4 Positionieren auf einen Inkrementalwert (DRVI)

					DRVI		FNC 158		
					Positionieren auf einen Inkrementalwert				CPU
						●	●		
Operanden	S1+	S2+	D1	D2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z,		Nur Y0 oder Y1	X, M, S			16 Bit	32 Bit	DRVI
						●	●	DDRVI	17

Funktion

Eine Position, die durch einen Inkrementalwert vorgegeben ist, wird mit einer festen Geschwindigkeit angefahren.

Beschreibung

- In S1+ wird die Anzahl der Inkremente vorgegeben. Der Wertebereich umfasst bei Wort-Operanden -32767 bis 32767 Inkremente und bei 32-Bit-Operanden -999999 bis 999999 Inkremente
- In S2+ wird die Frequenz der Ausgangsimpulse angegeben. Bei Wort-Operanden umfasst der Bereich 10 bis 32,767 Hz, bei 32-Bit-Operanden ist die Ausgabe einer Frequenz von 10 bis 100 kHz möglich.
- Zur Ausgabe der Impulse können in D1 nur die Ausgänge Y0 und Y1 angegeben werden. Da die Impulse mit hoher Frequenz ausgegeben werden, müssen Steuerungen mit Transistorausgängen verwendet werden. Relaiskontakte nutzen sich in diesem Fall vorzeitig ab und sind daher ungeeignet.
Zur Erzeugung steilflankiger Ausgangssignale sollte der Laststrom der Transistorausgänge mindestens 200 mA betragen. Eventuell ist die Verwendung von Pull-up-Widerständen notwendig.
- In D2+ wird ein Operand angegeben, der dem Servoverstärker die Drehrichtung vorgibt. Wenn der Operand nicht gesetzt ist, ist Linkslauf angewählt, bei gesetztem Operanden Rechtslauf.
- Bei der Positionierung auf einen Inkrementalwert wird die Richtung durch das Vorzeichen der Inkremente vorgegeben.
- Wenn während der Ausführung der Anweisung der Inhalt eines Operanden verändert wird, wird die Positionierung nicht beeinflusst. Die veränderten Operanden werden bei der nächsten Ausführung der Anweisung berücksichtigt.
- Wenn die Startbedingung der DRVI-Anweisung während der Ausführung der Anweisung nicht mehr erfüllt ist, wird eine Verzögerungsrampe durchlaufen und der Antrieb gestoppt. In diesem Fall wird der Sondermarker M8029, der das Ende der Bearbeitung anzeigt, nicht gesetzt.
- Nach dem Rücksetzen der Startbedingung kann die DRVI-Anweisung erst erneut gestartet werden, wenn die Sondermarker M8147 (Überwachung von Y0) und M8148 (Überwachung von Y0) nicht mehr gesetzt sind.
- Die minimale Ausgangsfrequenz kann mit der folgenden Formel berechnet werden:

$$f_{min} = \sqrt{\frac{f_{max}}{2 \times t_B}}$$

f_{max} ist die in D8146 und D8147 gespeicherte maximale Frequenz der Ausgangsimpulse.
 t_B ist Beschleunigungs- und Verzögerungszeit in der Einheit Sekunden

- Die folgenden Operanden stehen für Positionieraufgaben zur Verfügung:

Operand	Beschreibung
D8145	Drehzahl-Offset bei der Verwendung von FNC158 oder FNC159
D8147 (höherwertiges Wort) D8146 (niederwertiges Wort)	Maximale Frequenz der Ausgangsimpulse bei der Verwendung von FNC158 oder FNC159. Bereich: 100 Hz bis 100000 Hz
D8148	Beschleunigungs- und Verzögerungszeit [ms] bei der Verwendung von FNC158 oder FNC159
M8145	Impulsausgabe an Y0 sofort stoppen
M8146	Impulsausgabe an Y1 sofort stoppen
M8147	Überwachung der Impulsausgabe an Y0 (BUS/READY)
M8148	Überwachung der Impulsausgabe an Y1 (BUS/READY)

Tab. 7-19: Operanden zur Positionierung

HINWEIS

Beachten Sie bei der Programmierung der Startbedingung die zeitlichen Abläufe.

Beispiel ▾

DRVI-Anweisung

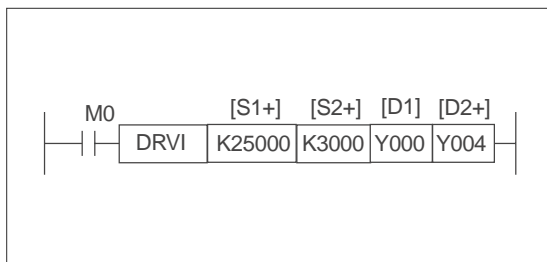


Abb. 7-66:

Programmbeispiel zum Einsatz der DRVI-Anweisung

C000403C



7.6.5 Positionieren auf einen Absolutwert (DRVA)

					DRVA		FNC 159		
					Positionieren auf einen Absolutwert				
					CPU	FX1S	FX1N	FX2N	
					Operanden S1+ S2+ D1 D2+ K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z, Nur Y0 oder Y1 X, M, S				
Puls-Anweisung (P)	Verarbeitung		Programmschritte						
						16 Bit	32 Bit	DRVA	9
						●	●	DDRVA	17

Funktion

Eine Position, die durch einen Absolutwert vorgegeben ist, wird mit einer festen Geschwindigkeit angefahren. Bei dieser Art der Positionierung wird ein Nullpunkt festgelegt. Die aktuelle Ist-Position wird als absoluter Wert in Bezug zu diesem Nullpunkt gemessen.

Beschreibung

- In S1+ wird die absolute Position vorgegeben. Der Wertebereich geht bei Wort-Operanden von -32767 bis 32767 und bei 32-Bit-Operanden von -999999 bis 999999.
- In S2+ wird die Frequenz der Ausgangsimpulse angegeben. Bei Wort-Operanden umfasst der Bereich 10 bis 32,767 Hz, bei 32-Bit-Operanden ist die Ausgabe einer Frequenz von 10 bis 100 kHz möglich.
- Zur Ausgabe der Impulse können in D1 nur die Ausgänge Y0 und Y1 angegeben werden. Da die Impulse mit hoher Frequenz ausgegeben werden, müssen Steuerungen mit Transistorausgängen verwendet werden. Relaiskontakte nutzen sich in diesem Fall vorzeitig ab und sind daher ungeeignet.
Zur Erzeugung steilflankiger Ausgangssignale sollte der Laststrom der Transistorausgänge mindestens 200 mA betragen. Eventuell ist die Verwendung von Pull-up-Widerständen notwendig.
- In D2+ wird ein Operand angegeben, der dem Servoverstärker die Drehrichtung vorgibt. Wenn der Operand nicht gesetzt ist, ist Linkslauf angewählt, bei gesetztem Operanden Rechtslauf.
- Bei der Positionierung auf einen Absolutwert wird als Ziel der Abstand vom Nullpunkt vorgegeben.
- Wenn während der Ausführung der Anweisung der Inhalt eines Operanden verändert wird, wird die Positionierung nicht beeinflusst. Die veränderten Operanden werden bei der nächsten Ausführung der Anweisung berücksichtigt.
- Wenn die Startbedingung der DRVI-Anweisung während der Ausführung der Anweisung nicht mehr erfüllt ist, wird eine Verzögerungsrampe durchlaufen und der Antrieb gestoppt. In diesem Fall wird der Sondermarker M8029, der das Ende der Bearbeitung anzeigt, nicht gesetzt.
- Nach dem Rücksetzen der Startbedingung kann die DRVI-Anweisung erst erneut gestartet werden, wenn die Sondermarker M8147 (Überwachung von Y0) und M8148 (Überwachung von Y0) nicht mehr gesetzt sind.
- Die minimale Ausgangsfrequenz kann mit der folgenden Formel berechnet werden:

$$f_{min} = \sqrt{\frac{f_{max}}{2 \times t_B}}$$

f_{max} ist die in D8146 und D8147 gespeicherte maximale Frequenz der Ausgangsimpulse.
 t_B ist Beschleunigungs- und Verzögerungszeit in der Einheit Sekunden

- Die folgenden Operanden stehen für Positionieraufgaben zur Verfügung:

Operand	Beschreibung
D8145	Drehzahl-Offset bei der Verwendung von FNC158 oder FNC159
D8147 (höherwertiges Wort) D8146 (niederwertiges Wort)	Maximale Frequenz der Ausgangsimpulse bei der Verwendung von FNC158 oder FNC159. Bereich: 100 Hz bis 100 kHz
D8148	Beschleunigungs- und Verzögerungszeit [ms] bei der Verwendung von FNC158 oder FNC159
M8145	Impulsausgabe an Y0 sofort stoppen
M8146	Impulsausgabe an Y1 sofort stoppen
M8147	Überwachung der Impulsausgabe an Y0 (BUS/READY)
M8148	Überwachung der Impulsausgabe an Y1 (BUS/READY)

Tab. 7-20: Operanden zur Positionierung

HINWEIS

Beachten Sie bei der Programmierung der Startbedingung die zeitlichen Abläufe.

Beispiel ▾

DRVA-Anweisung

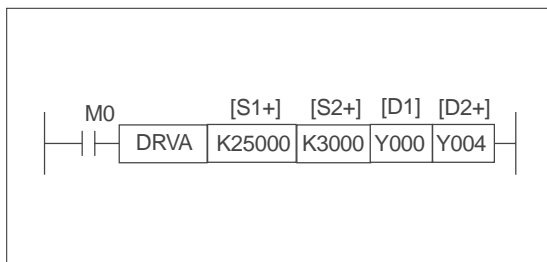


Abb. 7-67

Programmbeispiel zum Einsatz der DRVA-Anweisung

C000404C



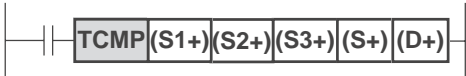
7.7 Echtzeituhr-Anweisungen

Übersicht der Anweisungen FNC 160 bis 167

Symbol	FNC	Bedeutung	Abschnitt
TCMP	160	Vergleich von Uhr-Daten	7.7.1
TZCP	161	Vergleich von Uhr-Daten mit einem Bereich	7.7.2
TADD	162	Addition von Uhr-Daten	7.7.3
TSUB	163	Subtraktion von Uhr-Daten	7.7.4
TRD	166	Lesen von Uhr-Daten	7.7.5
TRW	167	Schreiben von Uhr-Daten	7.7.6
HOUR	169	Betriebsstundenzähler	7.7.7

Tab. 7-23: Übersicht der Anweisung FNC 160 bis 167

7.7.1 Vergleich von Uhr-Daten (TCMP)

						TCMP		FNC 160		
						Vergleich von Uhr-Daten				
						CPU	FX1S	FX1N	FX2N	
							●	●	●	
Operanden	S1+	S2+	S3+	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z			T, C, D	Y, M, S		●	16 Bit	32 Bit	TCMP
				Es werden jeweils 3 aufeinanderfolgende Operandenadressen verwendet		●			TCMPP	11

Funktion

Vergleichen von Uhr-Daten mit Ausgabe der Vergleichsergebnisse

Beschreibung

- Die Uhr-Daten Stunde (S1+), Minute (S2+) und Sekunde (S3+) werden mit den in (S+) bis ((S+)+2) gespeicherten Uhr-Daten verglichen.
- Die Vergleichsergebnisse werden in 3 aufeinanderfolgenden Bit-Operanden gespeichert.
- Sind die Uhr-Daten in (S+) bis ((S+)+2) kleiner als die Uhr-Daten in (S1+) bis (S3+), wird der Bit-Operand (D+) gesetzt.
- Sind die Uhr-Daten in (S+) bis ((S+)+2) gleich den Uhr-Daten in (S1+) bis (S3+), wird der Bit-Operand ((D+)+1) gesetzt.
- Sind die Uhr-Daten in (S+) bis ((S+)+2) größer als die Uhr-Daten in (S1+) bis (S3+), wird der Bit-Operand ((D+)+2) gesetzt.

HINWEISE

Die angesprochenen Ausgangsoperanden bleiben nach Abschalten der Ausführungsbedingung der TCMP-Anweisung gesetzt.

In den Operanden (S1+) und (S+) können die Werte 0 bis 23 (Stunden) eingegeben werden.

In den Operanden (S2+) und ((S+)+1) können die Werte 0 bis 59 (Minuten) eingegeben werden.

In den Operanden (S3+) und ((S+)+2) können die Werte 0 bis 59 (Sekunden) eingegeben werden.

Für einen Vergleich der aktuellen Uhr-Daten der Echtzeituhr können die Register D8015 (Stunden), D8014 (Minuten) und D8013 (Sekunden) als (S1+), (S2+) und (S3+) verwendet werden.

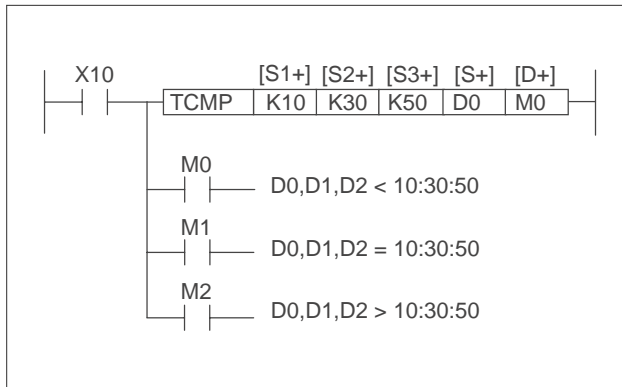
Beispiel ▾

Abb. 7-68:
 Programmierbeispiel zur TCMP-
 Anweisung

C000364C

Mit Setzen des Eingangs X10 werden die mit K10, K30 und K50 angegebenen 10 Stunden, 30 Minuten und 50 Sekunden mit den Uhr-Daten in D0 bis D2 verglichen.

Ist der Wert in D0 bis D2 kleiner als der Wert 10:30:50, wird der Merker M0 gesetzt.

Ist der Wert in D0 bis D2 gleich dem Wert 10:30:50, wird der Merker M1 gesetzt.

Ist der Wert in D0 bis D2 größer als der Wert 10:30:50, wird der Merker M2 gesetzt.

△

7.7.2 Vergleich von Uhr-Daten mit einem Bereich (TZCP)

					TZCP		FNC 161		
					Vergleich von Uhr-Daten mit einem Bereich				CPU
						●	●	●	
Operanden	S1+	S2+	S+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T, C, D ((S1+) ≤ (S2+))			Y, M, S			16 Bit	32 Bit	TZCP
	Es werden jeweils 3 aufeinanderfolgende Operandenadressen verwendet					●	●		TZCPP

Funktion

Vergleichen von Uhr-Daten mit einem Bereich mit Ausgabe der Vergleichsergebnisse

Beschreibung

- Die Uhr-Daten in (S+) bis ((S+)+2) werden mit den Uhr-Daten im Bereich zwischen (S1+) bis ((S1+)+2) und (S2+) bis ((S2+)+2) verglichen.
- Die Vergleichsergebnisse werden in 3 aufeinanderfolgenden Bit-Operanden gespeichert.
- Sind die Uhr-Daten in (S+) bis ((S+)+2) kleiner als die Uhr-Daten in (S1+) bis ((S1+)+2), wird der Bit-Operand (D+) gesetzt.
- Liegen die Uhr-Daten in (S+) bis ((S+)+2) in dem Bereich zwischen (S1+) bis ((S1+)+2) und (S2+) bis ((S2+)+2), wird der Bit-Operand ((D+)+1) gesetzt.
- Sind die Uhr-Daten in (S+) bis ((S+)+2) größer als die Uhr-Daten in (S2+) bis ((S2+)+2), wird der Bit-Operand ((D+)+2) gesetzt.

HINWEISE

Die angesprochenen Ausgangsoperanden bleiben nach Abschalten der Ausführungsbedingung der TCMP-Anweisung gesetzt.

In den Operanden (S1+), (S2+) und (S+) können die Werte 0 bis 23 (Stunden) eingegeben werden.

In den Operanden ((S1+)+1), ((S2+)+1) und ((S+)+1) können die Werte 0 bis 59 (Minuten) eingegeben werden.

In den Operanden ((S1+)+2), ((S2+)+2) und ((S+)+2) können die Werte 0 bis 59 (Sekunden) eingegeben werden.

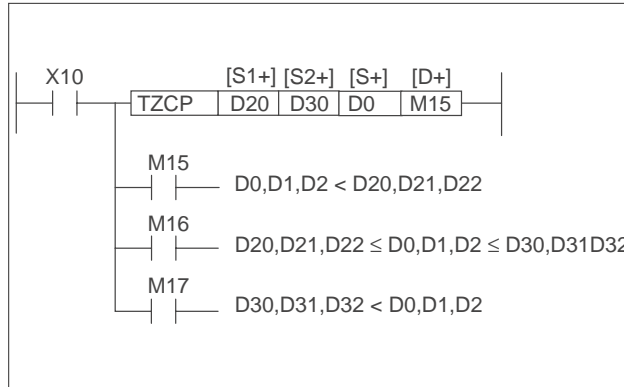
Beispiel ▾

Abb. 7-69:
 Programmierbeispiel zur TZCP-
 Anweisung

C000365C

Mit Setzen des Eingangs X10 werden die Uhr-Daten in D0 bis D2 mit dem Uhr-Datenbereich zwischen D20 bis D22 und D30 bis D32 verglichen.

Sind die Uhr-Daten in D0 bis D2 kleiner als die Uhr-Daten in D20 bis D22, wird der Merker M15 gesetzt.

Liegen die Uhr-Daten in D0 bis D2 in einem Bereich zwischen den Uhrdaten in D20 bis D22 und D30 bis D32, wird der Merker M16 gesetzt.

Sind die Uhr-Daten in D0 bis D2 größer als die Uhr-Daten in D30 bis D32, wird der Merker M17 gesetzt.

△

7.7.3 Addition von Uhr-Daten (TADD)

				TADD		FNC 162		
				Addition von Uhr-Daten				
				CPU	FX1S	FX1N	FX2N	
					●	●	●	
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T, C, D, Es werden jeweils 3 aufeinanderfolgende Operandenadressen verwendet				●	16 Bit	32 Bit	TADD
					●		TADDP	7

Funktion

Addieren von Uhr-Daten und speichern des Ergebnisses

Beschreibung

- Die Uhr-Daten in (S1+) bis ((S1+)+2) werden zu den Uhr-Daten in (S2+) bis ((S2+)+2) addiert. Das Ergebnis wird (D+) bis ((D+)+2) gespeichert.
- Die Berechnung erfolgt bezüglich der Überläufe (Sekunde - Minute und Minute - Stunde) fehlerfrei.

		[S1+]		[S2+]		[D+]	
Stunden	D10	10		D20	3	D30	13
Minuten	D11	30	+	D21	10	D31	41
Sekunden	D12	27		D22	49	D32	16
		10:30:27		03:10:49	⇒	13:41:16	

Abb. 7-70: Addition von Uhr-Daten

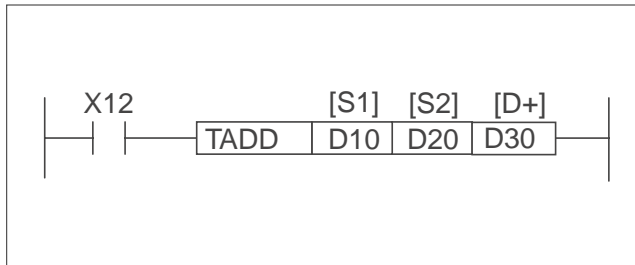
- Wird das Additionsergebnis größer als 24 Stunden, erfolgt eine Umschaltung auf 0 Stunden ("nächster Tag"), und das Carry-Flag M8022 wird gesetzt.

		[S1+]		[S2+]		[D+]	
Stunden	D10	10		D20	18	D30	4
Minuten	D11	17	+	D21	12	D31	30
Sekunden	D12	29		D22	34	D32	3
		10:17:29		18:12:34	⇒	04:30:03	

Abb. 7-71: Addition von Uhr-Daten mit Stundenüberlauf

- Ist das Additionsergebnis 0 (00:00:00, 0 Stunden, 0 Minuten, 0 Sekunden), wird das Zero-Flag M8020 gesetzt.

- Als Quelle und Ziel können dieselben Operanden ((S1+) bis ((S1+)+2), (S2+) bis ((S2+)+2)) verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus. Um eine einmalige Ausführung zu gewährleisten, sind gepulste Anweisungen oder Verriegelungen zu verwenden.

Beispiel ▾**Abb. 7-72:**

Programmierbeispiel zur TADD-Anweisung

C000367C

Mit Setzen des Eingangs X12 werden zu den Uhr-Daten in D10 bis D12 die Uhrdaten aus D20 bis D22 addiert. Das Ergebnis wird in D30 bis D32 gespeichert.

△

7.7.4 Subtraktion von Uhr-Daten (TSUB)

				TSUB		FNC 163			
				Subtraktion von Uhr-Daten				CPU	FX1S
Operanden	S1+	S2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte		
	T, C, D, Es werden jeweils 3 aufeinander folgende Operandenadressen verwendet				●	16 Bit	32 Bit	TSUB	7
						●		TSUBP	7

Funktion

Subtrahieren von Uhr-Daten und speichern des Ergebnisses

Beschreibung

- Die Uhr-Daten in (S2+) bis ((S2+)+2) werden von den Uhr-Daten in (S1+) bis ((S1+)+2) subtrahiert. Das Ergebnis wird in (D+) bis ((D+)+2) gespeichert.
- Die Berechnung erfolgt bezüglich der Unterläufe (Minute - Sekunde und Stunde - Minute) fehlerfrei.

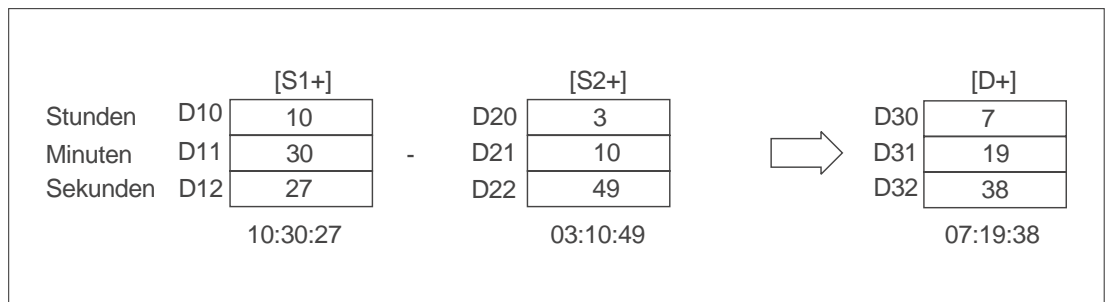


Abb. 7-73: Subtraktion von Uhr-Daten

- Wird das Subtraktionsergebnis kleiner als 0 Stunden (00:00:00), wird der Rest von 24 Stunden abgezogen ("vorhergehender Tag"), und das Borrow-Flag M8021 wird gesetzt.

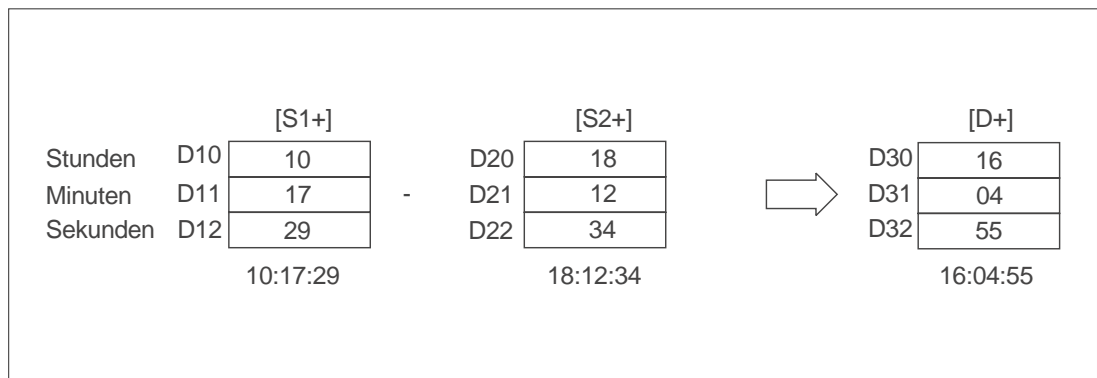
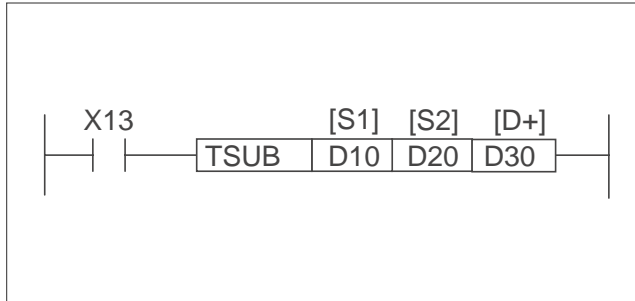


Abb. 7-74: Subtraktion von Uhr-Daten mit Stundenunterlauf

- Ist das Subtraktionsergebnis 0 (00:00:00, 0 Stunden, 0 Minuten, 0 Sekunden), wird das Zero-Flag M8020 gesetzt.

- Als Quelle und Ziel können dieselben Operanden ((S1+) bis ((S1+)+2), (S2+) bis ((S2+)+2)) verwendet werden. In diesem Fall wird das errechnete Ergebnis wieder in dem Quelloperanden gespeichert und anschließend für die nächste Berechnung genutzt. Dieser Prozeß wiederholt sich mit jedem Zyklus. Um eine einmalige Ausführung zu gewährleisten, sind gepulste Anweisungen oder Verriegelungen zu verwenden.

Beispiel ▾**Abb. 7-75:**

Programmierbeispiel zur TSUB-Anweisung

C000370C

Mit Setzen des Eingangs X13 werden die Uhr-Daten in D20 bis D22 von den Uhr-Daten in D10 bis D13 subtrahiert. Das Ergebnis wird in D30 bis D32 gespeichert.

△

7.7.5 Lesen von Uhr-Daten (TRD)

		TRD		FNC 166		
		Lesen von Uhr-Daten				
		CPU	FX1S	FX1N	FX2N	
			●	●	●	
Operanden	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	T, C, D, Es werden jeweils 7 aufeinanderfolgende Adressen des Operanden verwendet		●	16 Bit	32 Bit	TRD
			●		TRDP	5

Funktion

Lesen von Uhr-Daten

Beschreibung

- Mit der Anweisung erfolgt das Auslesen der Uhrdaten Jahr, Monat, Datum, Stunde, Minute, Sekunde und Wochentag aus der Echtzeituhr.
- Diese Daten werden in 7 aufeinanderfolgenden Operanden ab (D+) gespeichert.

Operand	Bedeutung	Wertebereich	⇒	Operand	Bedeutung
D8018	Jahr	00-99	⇒	D+	Jahr
D8017	Monat	01-12	⇒	(D+)+1	Monat
D8016	Datum	01-31	⇒	(D+)+2	Datum
D8015	Stunden	00-23	⇒	(D+)+3	Stunden
D8014	Minuten	00-59	⇒	(D+)+4	Minuten
D8013	Sekunden	00-59	⇒	(D+)+5	Sekunden
D8019	Wochentag	0-6 (Sonntag-Samstag)	⇒	(D+)+6	Wochentag

Tab. 7-24: Lesen von Uhr-Daten

HINWEISE

Die Jahreszahl wird als zweistelliger Wert eingelesen. Eine vierstellige Darstellung wird durch Speichern des Wertes 2000 im Register D8018 erreicht (siehe folgende Abb.).

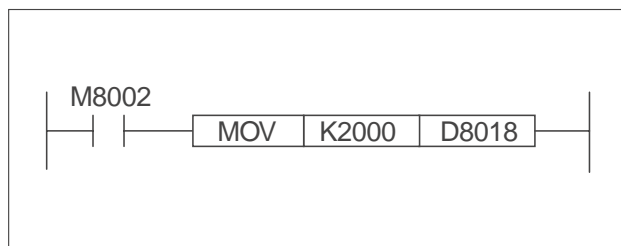


Abb. 7-76: Programmierbeispiel zum Beschreiben des Registers D8018

C000372C

Die vierstellige Darstellung der Jahreszahl ist erst nach der END-Verarbeitung des ersten Programmzyklus aktiv.

Die Bediengeräte FX-10DU-E und FX-20DU-E unterstützen die zweistellige Darstellung der Jahreszahl.

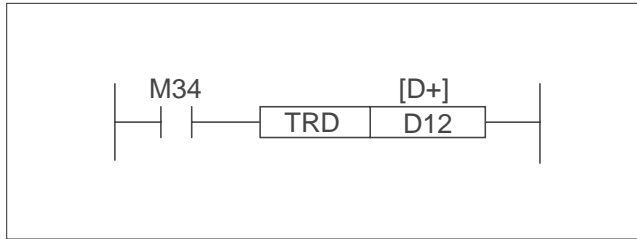
Beispiel ▾

Abb. 7-77:
Programmierbeispiel zur TRD-
Anweisung

C000373C

Mit Setzen des Merkers M34 werden die Uhr-Daten der Echtzeituhr ausgelesen und in den Registern D12 bis D18 gespeichert.

△

7.7.6 Schreiben von Uhr-Daten (TWR)

		TWR		FNC 167		
		Schreiben von Uhr-Daten				
Operanden		CPU	FX1S	FX1N	FX2N	
			●	●	●	
S+		Puls-Anweisung (P)	Verarbeitung		Programmschritte	
T, C, D, Es werden jeweils 7 aufeinanderfolgende Adressen des Operanden verwendet			●	16 Bit	32 Bit	TWR
				●		TWRP

Funktion

Schreiben von Uhr-Daten

Beschreibung

- Mit der Anweisung erfolgt das Schreiben der Uhrdaten Jahr, Monat, Datum, Stunde, Minute, Sekunde und Wochentag in die Echtzeituhr (D8013 bis D8019).
- Diese Daten sind in 7 aufeinanderfolgenden Operanden ab (S+) gespeichert.

Operand	Bedeutung	Wertebereich	⇒	Operand	Bedeutung
S+	Jahr	00-99	⇒	D8018	Jahr
(S+)+1	Monat	01-12	⇒	D8017	Monat
(S+)+2	Datum	01-31	⇒	D8016	Datum
(S+)+3	Stunden	00-23	⇒	D8015	Stunden
(S+)+4	Minuten	00-59	⇒	D8014	Minuten
(S+)+5	Sekunden	00-59	⇒	D8013	Sekunden
(S+)+6	Wochentag	0-6 (Sonntag-Samstag)	⇒	D8019	Wochentag

Tab. 7-25: Schreiben von Uhr-Daten

HINWEIS

Bei Verwendung der TWR-Anweisung ist das Setzen des Merkers M8015 (Anhalten der Echtzeituhr) nicht erforderlich.

Beispiel ▾

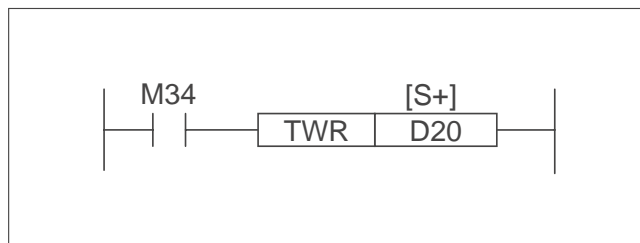


Abb. 7-78: Programmierbeispiel zur TWR-Anweisung

C000374C

Mit Setzen des Merkers M34 werden die Uhr-Daten aus den Registern D20 bis D26 in die Echtzeituhr geschrieben.

△

7.7.7 Betriebsstundenzähler (HOUR)

				HOUR		FNC 169									
				Betriebsstundenzähler											
<table border="1"> <tr> <th>Operanden</th> <th>S+</th> <th>D1+</th> <th>D2+</th> </tr> <tr> <td>K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z,</td> <td>D</td> <td>Z, Y, M, S</td> <td></td> </tr> </table>				Operanden	S+	D1+	D2+	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z,	D	Z, Y, M, S		CPU	FX1S	FX1N	FX2N
				Operanden	S+	D1+	D2+								
K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z,	D	Z, Y, M, S													
				●	●										
				Puls-Anweisung (P)	Verarbeitung		Programmschritte								
					16 Bit	32 Bit									
					●	●									

Funktion

Nach Ablauf einer einstellbaren Zeit wird ein Signal ausgegeben.

Beschreibung

- In (S+) wird die Zeit, in der Einheit Stunden, angegeben, nach der der in (D2+) angegebene Operand gesetzt wird.
- 16-Bit-Anweisung: (D1+) enthält die aktuelle Zahl der vollen Stunden. In ((D1+) + 1) wird die restliche Zeit in Sekunden abgelegt.
- 32-Bit-Anweisung: (D1+) und (D1+1) enthalten die Angabe der vollen Stunden. In ((D1+1) + 2) wird die restliche Zeit in Sekunden gespeichert.
- Der in (D2+) angegebene Operand wird eingeschaltet, wenn der Istwert in (D1+) größer als der Sollwert in (S+) ist.

HINWEISE

Um den Istwert der abgelaufenen Zeit auch bei Spannungsausfall zu sichern, sollte für (D1+) ein gepuffertes Register verwendet werden.

Die Zählung wird auch fortgesetzt, nachdem der in (D2+) angegebene Operand eingeschaltet wurde. Die Zählung wird angehalten, wenn der maximale Wertebereich des 16-Bit- oder 32-Bit-Operanden in (D1+) überschritten wird.

Wenn das Signal in (D2+) in festen Intervallen gesetzt werden soll, muss der Istwert in den Registern (D1+) bis ((D1+1) + 1) (16-Bit) bzw. (D1+) bis ((D1+1) + 2) (32-Bit) gelöscht werden, wenn der Sollwert erreicht wurde.

Beispiel ▾

HOUR-Anweisung

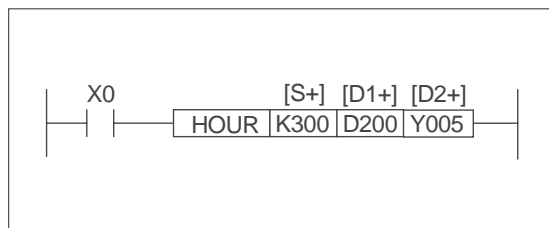


Abb. 7-79:

Programmbeispiel zum Einsatz der HOUR-Anweisung

C000405C

Nach Ablauf von 300 Stunden und einer Sekunde, nachdem X0 eingeschaltet wurde, wird Y5 eingeschaltet.



Beispiel ▾

DHOUR-Anweisung

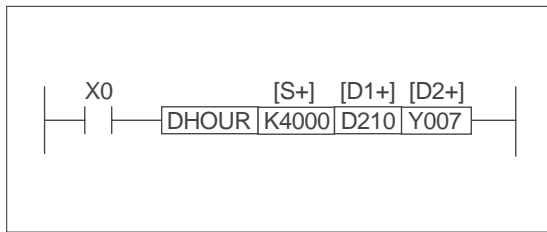


Abb. 7-80:
 Programmbeispiel zum Einsatz
 der DHOUR-Anweisung

C000406C

Zwischen dem Einschalten von X0 und Y7 vergehen 4000 Stunden und eine Sekunde.



7.8 Gray-Code-Anweisungen

Übersicht der Anweisungen FNC 170 bis 171

Symbol	FNC	Bedeutung	Abschnitt
GRY	170	Umwandlung Integer in Gray-Code	7.8.1
GBIN	171	Umwandlung Gray-Code in Integer	7.8.2

Tab. 7-26: Übersicht der Anweisung FNC 170 bis 171

7.8.1 Umwandlung Integer in Gray-Code (GRY)

		GRY		FNC 170	
		Umwandlung Integer in Gray-Code			
Operanden		S+		D+	
		K, H, KnX, KnY, KnM, KnS, T, C, D, V		KnY, KnM, KnS, T, C, D, V, Z	
		Puls-Anweisung (P)		Verarbeitung	
		●		16 Bit 32 Bit	
				● ●	
				Programmschritte	
				GRY, GRYP 5	
				DGRY, DGRYP 9	

Funktion

Konvertieren eines Integer-Wertes in den Gray-Code

Beschreibung

- Mit der Anweisung erfolgt das Konvertieren des Integer-Wertes ab (S+) in den Gray-Code.
- Das Ergebnis wird ab (D+) gespeichert.

HINWEIS

Durch die Charakteristik des Gray-Codes können numerische Werte durch inkrementieren der Quelldaten bei jedem Programmzyklus aktuell ausgegeben werden, ohne ein Stroboskop-Signal zu verwenden.

Beispiel ▾

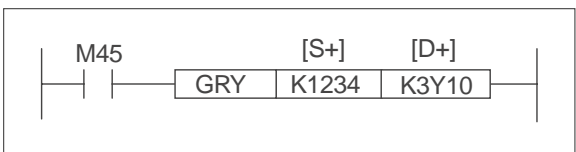


Abb. 7-81:
Programmierbeispiel zur GRY-Anweisung

C000375C

Mit Setzen des Merkers M45 wird der Integer-Wert K1234 in den Gray-Code konvertiert. Das Ergebnis wird an den Ausgängen Y10 bis Y23 ausgegeben.



7.8.2 Umwandlung Gray-Code in Integer (GBIN)

			GBIN		FNC 171					
			Umwandlung Gray-Code in Integer				CPU	FX1S	FX1N	FX2N
Operanden	S+		D+		Puls-Anweisung (P)		Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V		KnY, KnM, KnS, T, C, D, V, Z		●		16 Bit	32 Bit	GBIN, GBINP	
							●	●	DGBIN, DBBINP	

Funktion

Konvertieren eines Wertes im Gray-Code in einen Integer-Wert

Beschreibung

- Mit der Anweisung erfolgt das Konvertieren des im Gray-Code codierten Wertes ab (S+) in den Integer-Wert.
- Das Ergebnis wird ab (D+) gespeichert.

HINWEISE

Diese Anweisung kann zum Lesen von Daten aus einem Gray-Code-Codierer verwendet werden.

Wenn als Quelloperanden die Eingänge X0 bis X17 verwendet werden, lässt sich die Lesezeit durch Einstellung des Aktualisierungs-Filters (FNC51, REFF) verkürzen.

Beispiel ▾

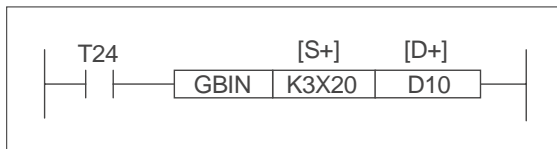


Abb. 7-82:
Programmierbeispiel zur GBIN-Anweisung

C000376C

Mit Setzen des Timerkontaktes T24 wird der Wert im Gray-Code an den Eingängen X20 bis X33 in einen Integer-Wert konvertiert. Das Ergebnis wird in D10 gespeichert.

△

7.9 Datenaustausch mit Analogmodul FX0N-3A (FX1N)

Symbol	FNC	Bedeutung	Abschnitt
RD3A	176	Lesen der analogen Eingangswerte	7.9.1
WR3A	177	Übertragen eines analogen Ausgangswertes	7.9.2

Tab. 7-21: Übersicht der Anweisungen zum Datenaustausch mit dem Analogmodul FX0N-3A

7.9.1 Lesen der analogen Eingangswerte (RD3A)

				RD3A		FNC 176		
				Lesen der analogen Eingangswerte				
				CPU	FX1S	FX1N	FX2N	
						●		
Operanden	n1+	n2+	D+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z				KnY, KnM, KnS, T, C, D, V, Z	16 Bit	32 Bit	RD3A
				●				

Funktion

Lesen der analogen Eingangswerte aus dem Analogmodul FX0N-3A

Beschreibung

- In (n1+) wird die Adresse des Sondermoduls angegeben (K0 bis K7).
- Der analoge Eingangswert des in (n2+) angegebenen Eingangskanals (K1 oder K2) wird in (D+) gespeichert.

HINWEIS | Vor dem Lesen der Analogwerte sollte die Eingangscharakteristik des Analogmoduls eingestellt werden. Nähere Hinweise hierzu finden Sie in der Bedienungsanleitung des Analogmoduls.

Beispiel ▾ RD3A-Anweisung

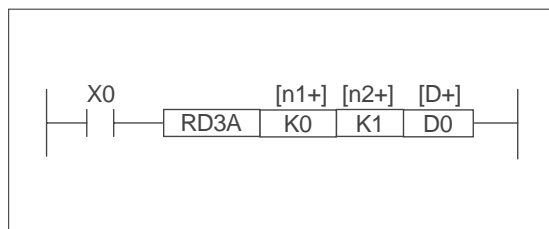


Abb. 7-83:
Programmbeispiel zum Einsatz der RD3A-Anweisung

C000407C

Der an Kanal 1 anstehende Analogwert des Analogmoduls mit der Adresse 0 wird in das Register D0 übertragen.



7.9.2 Scheiben eines analogen Ausgangswertes (WR3A)

				WR3A		FNC 177		
				Scheiben eines analogen Ausgangswertes				
Operanden				Puls-Anweisung (P)	Verarbeitung		Programmschritte	
					K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	16 Bit	32 Bit
●								

Funktion

Übertragen eines analogen Ausgangswertes an das Analogmodul FX0N-3A

Beschreibung

- In (n1+) wird die Adresse des Sondermoduls angegeben (K0 bis K7).
- Der analoge Ausgangswert (S+) wird an den in (n2+) angegebenen Ausgangskanal (nur K1) übertragen.

HINWEIS

Vor dem Übertragen des Ausgangswertes sollte die Ausgangscharakteristik des Analogmoduls eingestellt werden. Nähere Hinweise hierzu finden Sie in der Bedienungsanleitung des Analogmodul.

Beispiel ▾

WR3A-Anweisung

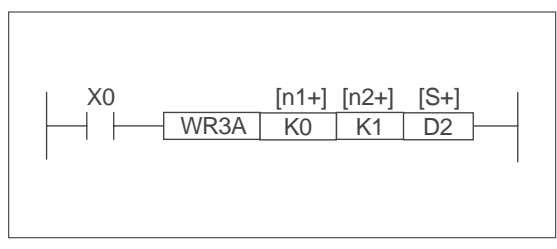


Abb. 7-84:
Programmbeispiel zum Einsatz der WR3A-Anweisung

C000408C

Der Inhalt des Registers D2 wird zur Ausgabe als analoger Wert an das Analogmodul mit der Adresse 0 übertragen.



7.10 Vergleichsanweisung II

Übersicht der Anweisungen FNC 224 bis 246

Symbol	FNC	Bedeutung	Abschnitt
LD =	224	Vergleichsanweisung, gleich	7.101
LD >	225	Vergleichsanweisung, größer	7.10.1
LD <	226	Vergleichsanweisung, kleiner	7.10.1
LD <>	228	Vergleichsanweisung, ungleich	7.10.1
LD ≤	229	Vergleichsanweisung, kleiner gleich	7.10.1
LD ≥	230	Vergleichsanweisung, größer gleich	7.10.1
AND =	232	UND-verknüpfte Vergleichsanweisung, gleich	7.10.2
AND >	233	UND-verknüpfte Vergleichsanweisung, größer	7.10.2
AND <	234	UND-verknüpfte Vergleichsanweisung, kleiner	7.10.2
AND <>	236	UND-verknüpfte Vergleichsanweisung, ungleich	7.10.2
AND ≤	237	UND-verknüpfte Vergleichsanweisung, kleiner gleich	7.10.2
AND ≥	238	UND-verknüpfte Vergleichsanweisung, größer gleich	7.10.2
OR =	240	ODER-verknüpfte Vergleichsanweisung, gleich	7.10.3
OR >	241	ODER-verknüpfte Vergleichsanweisung, größer	7.10.3
OR <	242	ODER-verknüpfte Vergleichsanweisung, kleiner	7.10.3
OR <>	244	ODER-verknüpfte Vergleichsanweisung, ungleich	7.10.3
OR ≤	245	ODER-verknüpfte Vergleichsanweisung, kleiner gleich	7.10.3
OR ≥	246	ODER-verknüpfte Vergleichsanweisung, größer gleich	7.10.3

Tab. 7-27: Übersicht der Anweisung FNC 224 bis 246

7.10.1 Lade Vergleiche (LD□)

		LD□		FNC 224 – 230			
		Lade Vergleiche					
		CPU	FX1S	FX1N	FX2N		
			●	●	●		
Operanden	S1+	S2+	Puls-Anweisung (P)		Verarbeitung	Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z			16 Bit ●	32 Bit ●	LD□ 5

Funktion

Laden der Vergleichsergebnisse LD□

Beschreibung

- Mit der Anweisung erfolgt das Laden der Vergleichsergebnisse der ab (S1+) und (S2+) angegebenen Werte.
- Ist das Vergleichsergebnis wahr, wird der LD-Kontakt gesetzt.
- Ist das Vergleichsergebnis falsch, wird der LD-Kontakt nicht gesetzt.
- Das □ in der LD□-Anweisung steht als Platzhalter für die Vergleichsoperationen =, >, <, <>, ≤ und ≥. Die folgende Tabelle gibt die Zuordnung der Vergleichsoperationen zu den entsprechenden Funktionsnummern wieder.

FNC Nummer	Anweisungen		Wahr wenn	Falsch wenn
	16 Bit	32 Bit		
224	LD =	DLD =	(S1+) = (S2+)	(S1+) <> (S2+)
225	LD >	DLD >	(S1+) > (S2+)	(S1+) ≤ (S2+)
226	LD <	DLD <	(S1+) < (S2+)	(S1+) ≥ (S2+)
228	LD <>	DLD <>	(S1+) <> (S2+)	(S1+) = (S2+)
229	LD ≤	DLD ≤	(S1+) ≤ (S2+)	(S1+) > (S2+)
230	LD ≥	DLD ≥	(S1+) ≥ (S2+)	(S1+) < (S2+)

Tab. 7-28: Übersicht der LD□-Anweisungen

HINWEIS | Die LD□-Anweisung kann wie eine LD-Anweisung verwendet werden.

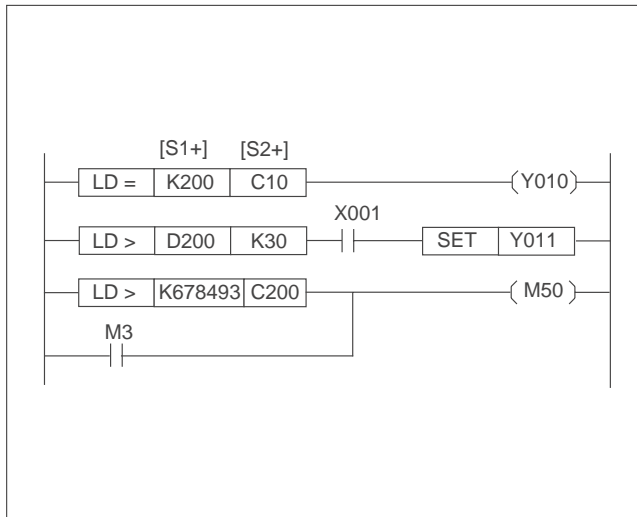
Beispiel ▾

Abb. 7-85:
 Programmierbeispiel zu den LD□-
 Anweisungen

C000377C

Ist der Wert K200 gleich dem Counter-Wert C10, wird der Ausgang Y10 gesetzt.

Ist der Wert in D200 größer als der Wert K30 und der Eingang X1 eingeschaltet, wird der Ausgang Y11 über die SET-Anweisung gesetzt.

Ist der Wert K678493 größer als der Counter-Wert C200 oder der Merker M3 gesetzt, wird der Merker M50 gesetzt.

△

7.10.2 UND-verknüpfte Vergleiche (AND□)

			AND□		FNC 230 – 238		
			UND-verknüpfte Vergleiche				
			CPU	FX1S	FX1N	FX2N	
			●	●	●		
Operanden	S1+	S2+	Puls-Anweisung (P)	Verarbeitung		Programmschritte	
	K, H, KnX, KnY, KnM, KnS, T, C, D, V	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		16 Bit	32 Bit	AND□	5
				●	●	DAND□	9

Funktion

UND-verknüpfen eines Vergleichsergebnisses

Beschreibung

- Mit der Anweisung erfolgt eine UND-Verknüpfung der Vergleichsergebnisse der ab (S1+) und (S2+) angegebenen Werte.
- Ist das Vergleichsergebnis wahr, wird der UND-verknüpfte Kontakt gesetzt.
- Ist das Vergleichsergebnis falsch, wird der UND-verknüpfte Kontakt nicht gesetzt.
- Das □ in der AND□-Anweisung steht als Platzhalter für die Vergleichsoperationen =, >, <, <>, ≤ und ≥. Die folgende Tabelle gibt die Zuordnung der Vergleichsoperationen zu den entsprechenden Funktionsnummern wieder.

FNC Nummer	Anweisungen		Wahr wenn	Falsch wenn
	16 Bit	32 Bit		
232	AND =	DAND =	(S1+) = (S2+)	(S1+) <> (S2+)
233	AND >	DAND >	(S1+) > (S2+)	(S1+) ≤ (S2+)
234	AND <	DAND <	(S1+) < (S2+)	(S1+) ≥ (S2+)
236	AND <>	DAND <>	(S1+) <> (S2+)	(S1+) = (S2+)
237	AND ≤	DAND ≤	(S1+) ≤ (S2+)	(S1+) > (S2+)
238	AND ≥	DAND ≥	(S1+) ≥ (S2+)	(S1+) < (S2+)

Tab. 7-29: Übersicht der AND□-Anweisung

HINWEIS | Die AND□-Anweisung kann wie eine AND-Anweisung verwendet werden.

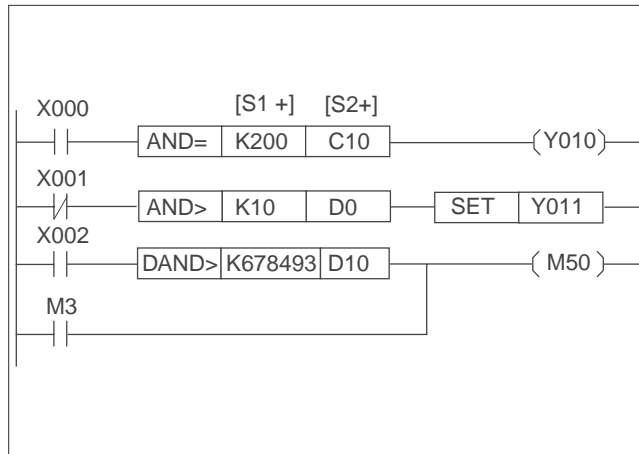
Beispiel ▾

Abb. 7-86:
 Programmierbeispiel zu den AND□-
 Anweisungen

C000378C

Ist der Wert K200 gleich dem Counter-Wert C10 und der Eingang X0 eingeschaltet, wird der Ausgang Y10 gesetzt.

Ist der Wert K10 größer als der Wert in D0 und der Eingang X1 nicht eingeschaltet, wird der Ausgang Y11 über die SET-Anweisung gesetzt.

Ist der Wert K678493 größer als der Wert in D10 und D11 und der Eingang X2 eingeschaltet, wird der Merker M50 gesetzt. Der Merker M50 wird auch gesetzt, wenn M3 gesetzt wird.

△

7.10.3 ODER-verknüpfte Vergleiche (OR□)

		OR□		FNC 240 – 246	
		ODER-verknüpfte Vergleiche			
		CPU	FX1S	FX1N	FX2N
			●	●	●
Operanden	S1+	S2+	Puls-Anweisung (P)		Verarbeitung
	K, H, KnX, KnY, KnM, KnS, T, C, D, V	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		16 Bit	32 Bit
				●	●
				Programmschritte	
				OR□	5
				DOR□	9

Funktion

ODER-verknüpfen eines Vergleichsergebnisses

Beschreibung

- Mit der Anweisung erfolgt eine ODER-Verknüpfung der Vergleichsergebnisse der ab (S1+) und (S2+) angegebenen Werte.
- Ist das Vergleichsergebnis wahr, wird der ODER-verknüpfte Kontakt gesetzt.
- Ist das Vergleichsergebnis falsch, wird der ODER-verknüpfte Kontakt nicht gesetzt.
- Das □ in der OR□-Anweisung steht als Platzhalter für die Vergleichsoperationen =, >, <, <>, ≤ und ≥. Die folgende Tabelle gibt die Zuordnung der Vergleichsoperationen zu den entsprechenden Funktionsnummern wieder.

FNC Nummer	Anweisungen		Wahr wenn	Falsch wenn
	16 Bit	32 Bit		
240	OR =	DOR =	(S1+) = (S2+)	(S1) <> (S2+)
241	OR >	DOR >	(S1+) > (S2+)	(S1) ≤ (S2+)
242	OR <	DOR <	(S1+) < (S2+)	(S1) ≥ (S2+)
244	OR <>	DOR <>	(S1+) <> (S2+)	(S1) = (S2+)
245	OR ≤	DOR ≤	(S1+) ≤ (S2+)	(S1) > (S2+)
246	OR ≥	DOR ≥	(S1+) ≥ (S2+)	(S1) < (S2+)

Tab. 7-22: Übersicht der OR□-Anweisung

HINWEIS | Die OR□-Anweisung kann wie eine OR-Anweisung verwendet werden.

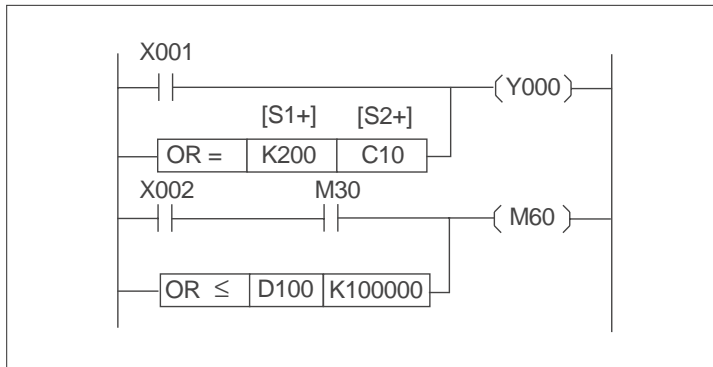
Beispiel ▾

Abb. 7-87:
 Programmierbeispiel zu den
 OR□-Anweisungen

C000379C

Ist der Wert K200 gleich dem Counter-Wert C10 oder der Eingang X1 eingeschaltet, wird der Ausgang Y0 gesetzt.

Ist der Wert in D100 kleiner oder gleich dem Wert K100000 oder sind der Eingang X2 und der Merker M30 gesetzt, wird der Merker M60 gesetzt.

△

8 Sonderfunktionen

Die Steuerungen der FX-Familie stellen einige Sonderfunktionen zur Verfügung, mit denen Sie die Einsatzmöglichkeiten der Steuerung erweitern können. Diese Sonderfunktionen sind deshalb in einem eigenen Kapitel zusammengefaßt, weil sie nicht unmittelbar durch eine bestimmte Anweisung ausgeführt werden.

In diesem Kapitel werden folgende Sonderfunktionen beschrieben:

- Datenerhalt im STOP-Modus
- Betrieb mit konstanter Programmzykluszeit
- Passwortfunktion
- Pulse-Catch-Funktion
- Eingangfilter einstellen
- Integrierte Potentiometer bei FX1S und FX1N
- Echtzeituhr
- File-Register
- RUN/STOP-Umschaltung
- 24 V DC-Grundgeräte
- Anzeigemodul FX1N-5DM für FX1S und FX1N

8.1 Datenerhalt im STOP-Modus

Im „normalen“ Betriebsablauf setzen Steuerungen der FX-Familie alle Ausgangssignalzustände auf „0“, sobald die Steuerung vom RUN-Modus in den STOP-Modus geschaltet wird. Für einige Anwendungen ist es jedoch sinnvoll, auch im STOP-Modus die zuletzt vorhandenen Ausgangssignalzustände aufrecht zu erhalten. Dies können Sie erreichen, wenn Sie den Sondermerker M8033 im SPS-Programm setzen. Die Istwerte von Timern und Countern bleiben dann ebenfalls erhalten.

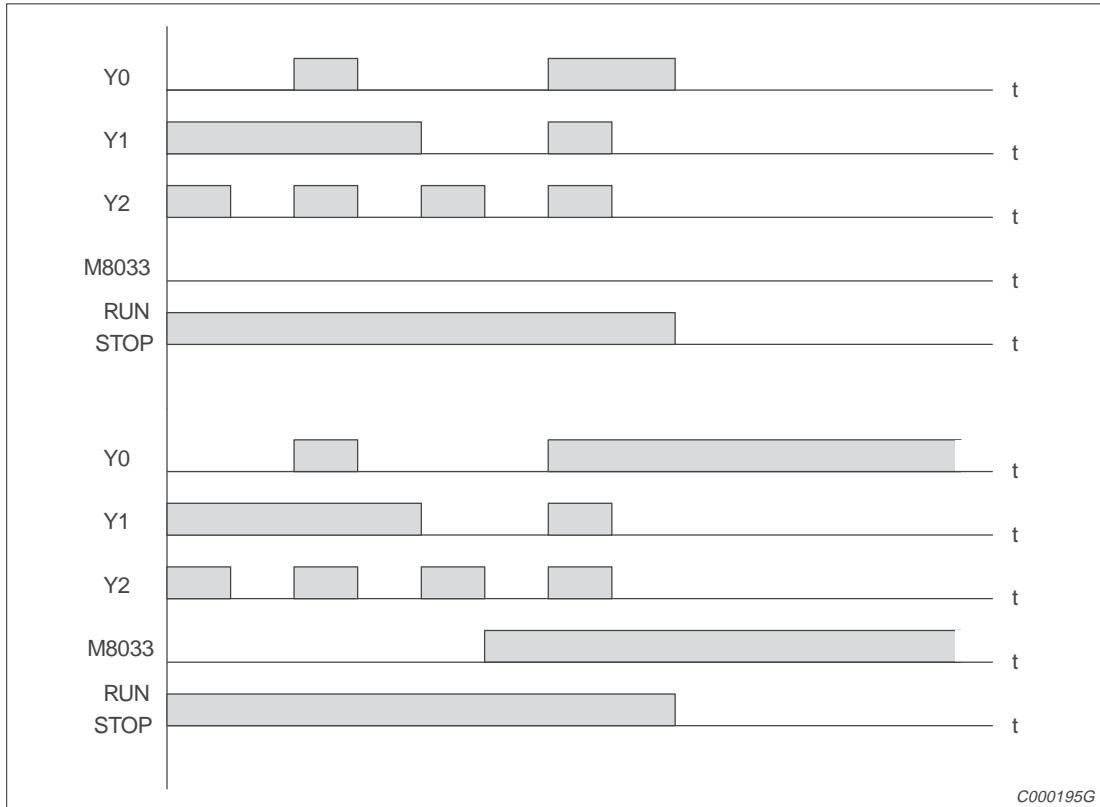


Abb. 8-1: Beispiel für Datenerhalt im STOP-Modus

Das folgende Beispiel zeigt die notwendige Programmierung.

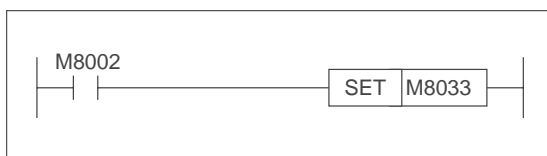


Abb. 8-2: Programmierbeispiel

C000193C

8.2 Betrieb mit konstanter Programmzykluszeit

Die Steuerungen der FX-Familie können mit einer konstanten vom SPS-Programm unabhängigen Programmzykluszeit betrieben werden. Dies ist z. B. beim Einsatz der RAMP-Applikationsanweisung notwendig.

Um eine konstante Programmzykluszeit zu erreichen, muß der Sondermerker M8039 im SPS-Programm gesetzt werden. Die Programmzykluszeit kann in Einheiten von 1ms festgelegt werden. Den gewählten Programmzykluszeitwert müssen Sie ins Datenregister D8039 schreiben. Wählen Sie die Programmzykluszeit größer als die durchschnittliche Programmzykluszeit. Der Wert der durchschnittlichen Programmzykluszeit wird von der SPS automatisch im Datenregister D8010 abgespeichert.

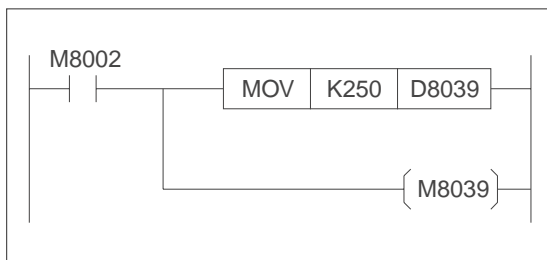


Abb. 8-3:

Festlegen einer konstanten Programmzykluszeit auf den Wert 250 ms im Datenregister D8039

C000194C

8.3 Passwortfunktion

Durch Einsatz der Passwortfunktion kann ein abgespeichertes SPS-Programm gegen unerwünschten Zugriff gesichert werden.

Die in der Tabelle 9-1 angegebenen 3 Schutzebenen stehen zur Verfügung.

Schutzebene, Kennbuchstabe	Bedeutung	Mögliche Zugriffe				
		Programm testen	Monitorfunktion	Programm lesen	Programm schreiben	Sollwerte verändern
A	Schutz gegen alle Zugriffe	—	—	—	—	—
B	Kopierschutz	●	●	—	—	—
C	Datenschutz	●	●	●	—	●

Tab. 8-1: Schutzebenen der Passwortfunktion

Das Passwort wird über das jeweilige Programmiersystem eingegeben. Das Passwort besteht aus einem Kennbuchstaben und einem siebenstelligen hexadezimalen Code.

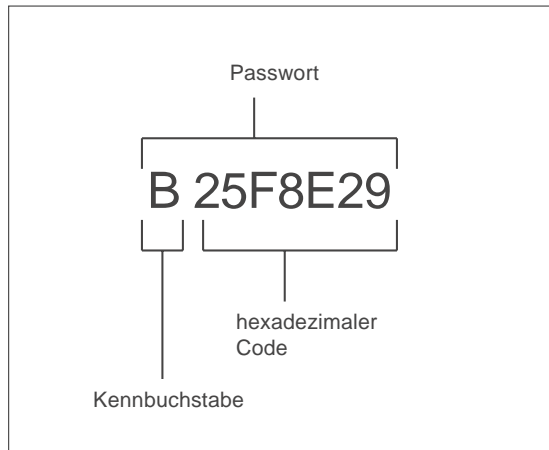


Abb. 8-4: Eingeben des Passwortes

C000197G

HINWEISE

Beachten Sie, daß an einem SPS-Programm, dessen Passwort nicht mehr bekannt ist, keine Änderungen mehr möglich sind. Das SPS-Programm können Sie nur noch komplett löschen.

Durch ein Passwort mit der Schutzebene A wird auch der Zugriff durch eine Bedieneinheit wie z. B. FX-20 DU verhindert. Möchten Sie einen Zugriff durch diese Bedieneinheit ermöglichen, müssen Sie die Schutzebene B wählen.

Bei der Festlegung eines Passwortes mit der MELSEC MEDOC-Software ist zu beachten, daß nur die Schutzebene B eingesetzt werden kann.

Bei Ausführungen ab Versionsnummer V 1.63 von MELSEC MEDOC sind alle drei Schutzebenen möglich.

8.4 Pulse-Catch-Funktion

Mit der Pulse-Catch-Funktion können sehr kurzzeitige Eingangssignalimpulse, z. B. Lichtschrankensignale, von der Steuerung verarbeitet werden. Die minimale Impulslänge, die noch von der Steuerung verarbeitet werden kann, beträgt 300µs.

Die Pulse-Catch-Funktion kann nur für Signale benutzt werden, die über die Eingänge X0, X1, X2, X3, X4 und X5 in die Steuerung eingespeist werden. Es kann in jedem Programmzyklus jeweils nur 1 Impuls verarbeitet werden.

Bei jedem auflaufenden Impuls an einem der Eingänge wird automatisch von der Steuerung ein Sondermerker gesetzt. Dieser Sondermerker kann dann im Programm weiterverarbeitet werden. Damit die Steuerung einen neuen Impuls an einem Eingang erkennen kann, muß der zugehörige Sondermerker vorher im Programm zurückgesetzt werden.

Eingang	X0	X1	X2	X3	X4	X5
Sondermerker	M8170	M8171	M8172	M8173	M8174	M8175

Tab. 8-2: Eingänge und zugehörige Sondermerker

HINWEISE

Die Pulse-Catch-Funktion ist keine High-Speed-Funktion. Es kann in jedem Programmzyklus jeweils nur 1 Eingangsimpuls verarbeitet werden.

Beachten Sie, daß die Eingänge X0 bis X5 nicht gleichzeitig als Interrupt-Eingänge für die Pulse-Catch-Funktion und als Zählengänge für High-Speed-Counter eingesetzt werden können. Eine Doppelbelegung von Eingängen ist nicht zulässig.

Die Pulse-Catch-Funktion benötigt einen aktiven EI-Befehl (FNC04) im Programm.

Beispiel ▾

Einsatz der Pulse-Catch-Funktion zum Zählen von Impulsen an einer Lichtschranke über den Eingang X3.

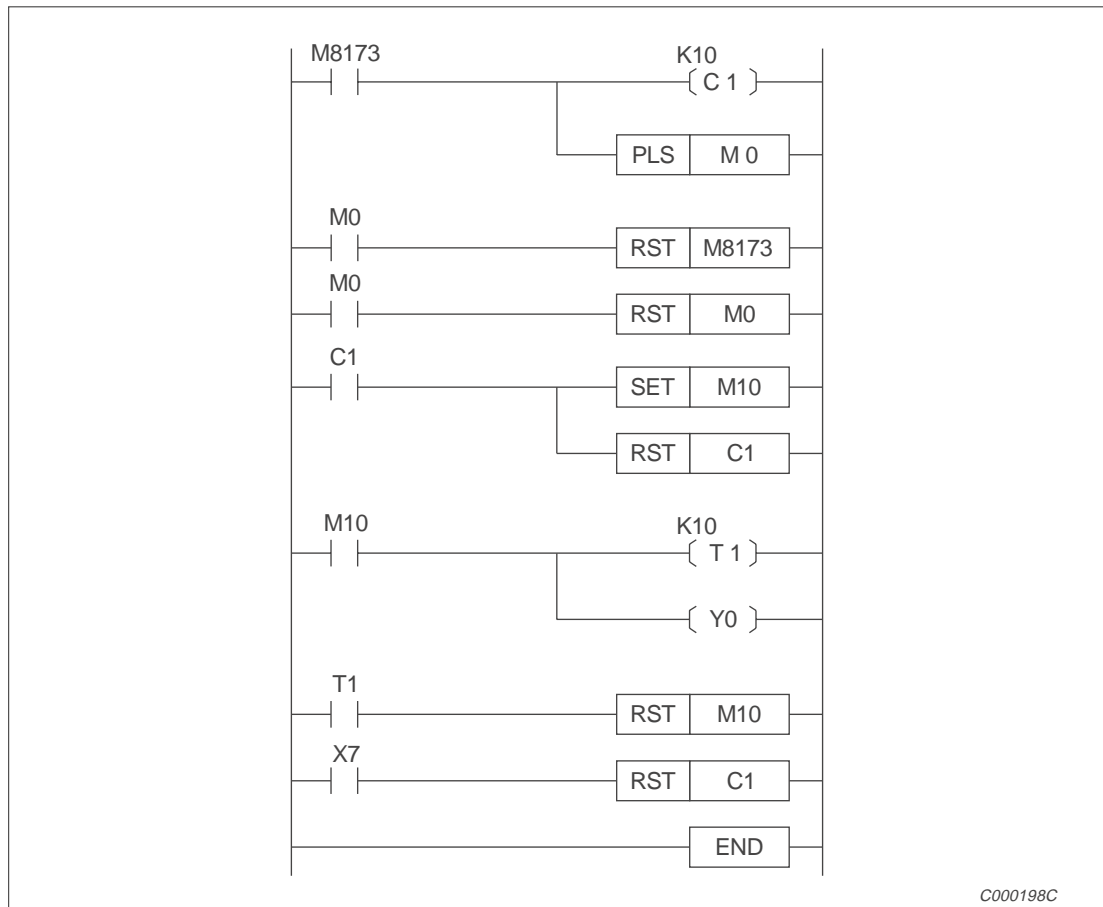


Abb. 8-5: Programmierbeispiel für den Einsatz der Pulse-Catch-Funktion und das Zählen von Impulsen einer Lichtschranke über den Eingang X3

Nach 10 Impulsen wird der Ausgang Y0 für 1s gesetzt. Der Zähler C1 wird über den Eingang X7 zurückgesetzt.

△

8.5 EingangsfILTER einstellen

FX1S/FX1N

FX1S- und FX1N-Serie

Die Verzögerungszeit der Eingangssignalbearbeitung wird in der SPS in den Eingangsfiltern festgelegt. Die Verzögerungszeit läßt sich in 1-ms-Schritten von 0 bis 15 ms festlegen. Der Standardwert beträgt 10 ms.

Die Filterzeit der Eingänge X0 bis X7 wird im Datenregister D8020 festgelegt.

Bei jeder Umschaltung vom STOP-Modus in den RUN-Modus wird der Standardwert von 10 ms im Datenregister D8020 abgespeichert.

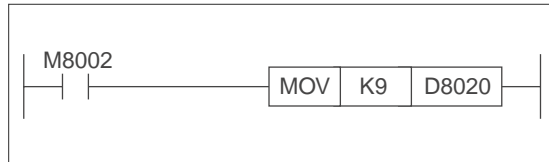


Abb. 8-6:

Einstellung der Verzögerungszeit der Eingangssignalverarbeitung für die Eingänge X0 bis X7 auf den Wert 9 ms im Datenregister D8020

HINWEISE

Die Verzögerungszeit läßt sich nur blockweise für die Eingänge X0 bis X7 einstellen.

Wenn der Wert "0" eingetragen wird, beträgt die Verzögerungszeit 10 μ s.

Wenn in einem Programm die Eingänge X0 bis X3 als Zählereingänge für High-Speed-Counter eingesetzt werden, wird automatisch die Verzögerungszeit auf den Wert 50 μ s eingestellt.

Wenn Sie sehr kurze Verzögerungszeiten einstellen (≤ 5 ms), müssen Sie gewährleisten, daß die Eingangssignale nicht durch Störungen beeinflußt werden. Hierdurch kann es zu Programmablauf Fehlern kommen.

FX2N

FX2N-Serie

Die EingangsfILTER werden über die Anweisung REFF (FNC51) eingestellt.

8.6 Integrierte Potentiometer bei FX1S und FX1N

Die Steuerungen MELSEC FX1S und FX1N verfügen über analoge Potentiometer zur Vorgabe von Zahlenwerten zwischen 0 bis 255.

Die Potentiometer sind direkt mit Datenregistern verbunden.

FX1S FX1N	Potentiometer 1 (VR1)	D8030
	Potentiometer 2 (VR2)	D8031

Tab. 8-3:
Datenregister der Potentiometer

Beispiel ▾

100-ms-Timer mit variabler Zeit

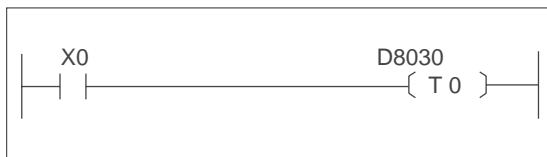


Abb. 8-7:
Timer mit Sollwert, der durch das Potentiometer VR1 verändert werden kann

C000196C

Wenn D8030 = 200, dann läuft eine Zeit von 20 s ab.

Wird eine kürzere oder eine kleiner abgestufte Zeit benötigt, kann ein 10-ms-Timer eingesetzt werden.

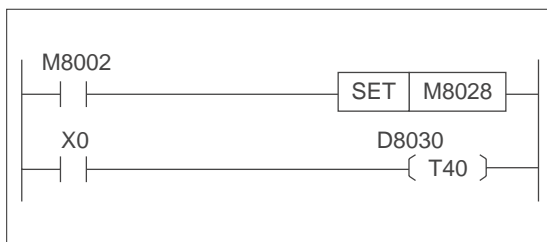


Abb. 8-8:
10-ms-Timer mit variabler Zeit (FX1S)

C000197C



8.7 Echtzeituhr-Funktion

Bei den Steuerungen der FX1S-, FX1N- und FX2N-Serie ist eine Echtzeituhr eingebaut. Die Uhr ist batteriegepuffert und läuft auch weiter, wenn die Versorgungsspannung der Steuerung abgeschaltet ist.

Funktionen

Die batteriegepufferte Echtzeituhr liefert Informationen zum Datum und zur Uhrzeit.

Schaltjahre werden in dem Zeitraum von 1980 bis 2079 berücksichtigt.

Sonderregister	Zeit	Einstellung
D8013	Sekunden	0 - 59
D8014	Minuten	0 - 59
D8015	Stunden	0 - 23
D8016	Tag	1 - 31
D8017	Monat	1 - 12
D8018	Jahr	0 - 99 (1980 - 1999; 2000 - 2079)
D8019	Wochentag	0 - 6 (Sonntag - Samstag)

Tab. 8-4:
Register für die Echtzeituhr

Genauigkeit

Genauigkeitsschwankungen liegen beim Einsatz des Geräts in einem Bereich von 25°C bei ± 45 s innerhalb eines Monats.

Sondermerker für den Betrieb einer Echtzeituhr

Sondermerker	Bedeutung	Beschreibung
M8015	Einstellung der Zeit	Ist M8015 eingeschaltet, stoppt die Uhr. Die Uhrzeit kann über das Programmiergerät eingegeben bzw. korrigiert werden. Wird M8015 wieder ausgeschaltet, beginnt die Uhrfunktion.
M8016	Datenerhalt	Ist M8016 eingeschaltet, bleiben die Daten in den entsprechenden Datenregistern erhalten.
M8017	Aufrunden der Minuten	Beim Einschalten von M8017 wird die Minutenangabe entsprechend auf- oder abgerundet.
M8018	Uhr aktiviert	M8018 ist automatisch gesetzt, um anzuzeigen, daß die Uhrfunktion aktiviert ist.
M8019	Setzfehler	M8019 wird eingeschaltet, wenn die eingegebenen Daten außerhalb des zulässigen Bereichs liegen.

Tab. 8-5: *Bedeutung der Sondermerker*

HINWEIS

Die Steuerungen der FX1S-, FX1N und FX2N-Serie verfügen über spezielle Befehle zum einfachen Stellen und Benutzen der Echtzeituhr (Nähere Informationen enthält Absatz 7.7).

8.8 File-Register

Unter File-Registern versteht man Register, die als zusätzliche, spannungsausfallgeschützte Datenspeicher eingesetzt werden können.

Alle Steuerungen der MELSEC FX1S, FX1N und FX2N-Serie verfügen über diese Register.

Zusätzliche Informationen finden Sie in der Beschreibung zum Operandensatz zu File-Registern im Anhang dieses Handbuchs.

Einrichten von File-Registern

Die File-Register müssen in Blöcken zu jeweils 500 Registern im Parametersatz der SPS eingetragen werden. Dies erfolgt mit einem Programmiersystem wie z.B. MELSEC MEDOC *plus*.

Lesen der File-Register durch das SPS-Program

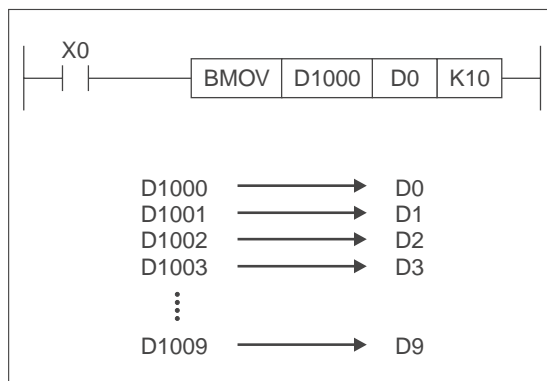


Abb. 8-9:
Lesen von File-Registern

C000195C

8.9 RUN-/STOP-Umschaltung

Bei der FX2N-Serie kann einer der Eingänge X0 bis X17 (X0 bis X7 bei der FX2N-16M) als RUN-Klemme parametrisiert werden.

Die Steuerung kann über eine Brücke zwischen der 24V- und der RUN-Klemme bzw. durch den RUN-/STOP-Schalter in den RUN-Modus geschaltet werden.

Zusätzlich besteht die Möglichkeit, bei unbeschalteter RUN-Klemme bzw. auf STOP geschaltetem RUN-/STOP-Schalter die Steuerung über 3 Sondermerker in den RUN- oder STOP-Modus zu stellen.

Merker	RUN	STOP
M8035	1	0
M8036	1	0
M8037	0	1

Tab. 8-6:
Belegung der Merker

Wenn M8037 eingeschaltet wird, werden die Merker M8035 und M8036 zurückgesetzt.

Die Merker können beispielsweise über ein Programmiersystem oder über ein Bediengerät geschaltet werden.

Bei Verwendung der FX1N- und FX2N-Serie ist die Umschaltung in den STOP-Betrieb auch dann möglich, wenn der RUN-Betrieb durch ein Terminal oder den RUN-/STOP-Schalter aktiviert ist. Die Umschaltung erfolgt durch Setzen des Merkers M8037. Nach Rücksetzen des Merkers kehrt die CPU in den RUN-Modus zurück.

8.10 FX2N-Grundgeräte mit 24 V DC-Versorgung

Um einen einwandfreien Betrieb dieser Geräte zu gewährleisten, muss folgende Programmzeile an den Anfang des SPS-Programms eingefügt werden.

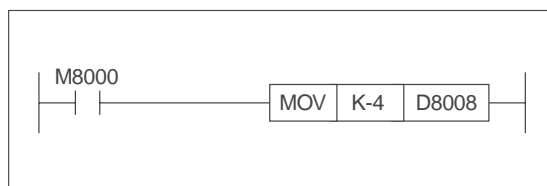


Abb. 8-10:
Programmierbeispiel

C000191C

Bei AC-gespeisten Grundgeräten der FX2N-Serie kann die Erfassung der Spannungsausfallzeit durch Eintrag des gewünschten Erfassungszeitraums in das Sonderregister D8008 vorgegeben werden.

Bei DC-gespeisten Modulen muss dieser Erkennungszeitraum auf 5 ms festgelegt werden. Dies wird durch den Eintrag des Wertes -4 in D8008 erreicht.

Ohne diesen Eintrag kann es bei Spannungsausfällen der DC-Versorgung zu fehlerhaften Erfassungsdaten kommen.

8.11 Anzeigemodul FX1N-5DM für FX1S und FX1N

Das Anzeigemodul FX1N-5DM kann zur Beobachtung der Zustände von Operanden und zur Eingabe von Daten bei den Steuerungen der FX1S- und FX1N-Serie eingesetzt werden.

8.11.1 Funktionen

Funktionen zur Bedienung

Die folgenden Funktionen können nur durch die Tasten am Anzeigemodul FX1N-5DM angewählt und ausgeführt werden. Nähere Hinweise zur Verwendung der Funktionstasten finden Sie in der Bedienungsanleitung des Anzeigemoduls.

Funktion		Beschreibung
Uhr	Anzeige	Anzeige der in der FX1S/FX1N eingebauten Echtzeituhr
	Stellen	Einstellen des Datums und der Uhrzeit
Beobachten von Operanden	Bit-Operanden	Anzeige des Zustandes von Eingängen (X), Ausgängen (Y), Merkern (M) und Sondermerkern (S)
	Word-Operanden (16 Bit)	Anzeige der Istwerte von Timern (T), Zählern (C) und Registern (D) Die Sollwerte der Timer und Zähler können eingestellt werden.
	Word-Operanden (32 Bit)	Anzeige der Istwerte von 32-Bit-Zählern und -Registern (D) Einstellung der Sollwerte der 32-Bit-Zähler
Beobachten des Pufferspeichers (nur bei FX1N)		Anzeige des Pufferspeicherinhaltes von Sondermodulen
Fehleranzeige		Anzeige von Fehlercodes und der Schrittnummer des Fehlers
Forcen von Operanden		Zwangswaises Ein- und Ausschalten von Bit-Operanden (X, M, S)
Rücksetzen von Timern und Zählern		Löschen des Istwertes von Timern und Zählern
Verändern von Daten	Istwerte	Änderung der Istwerte von Timern, Zählern und Register
	Sollwerte	Änderung der Sollwerte von Timern und Zählern

Tab. 8-7: Funktionen, die durch die Funktionstasten angewählt werden können

Funktionen zur Steuerung des Anzeigemoduls

Das Anzeigemodul kann durch das SPS-Programm gesteuert werden.

Funktion	Beschreibung
Schutz vor unbefugte Zugriffe	Wahlweise können alle Funktionen, nur Beobachtungsfunktionen oder nur die Anzeige von Datum und Uhrzeit freigegeben werden.
Festlegung der angezeigten Operanden	Der Anwender kann die Art und die Adresse der Operanden festlegen, die angezeigt werden.
Ein- und Ausschalten der Fehleranzeige	Wahlweise kann die Anzeige von Fehlercode ein- oder ausgeschaltet werden.
Automatisches Ausschalten der Beleuchtung der Anzeige	Einstellung der Zeit, nach der sich die Hintergrundbeleuchtung der Anzeige automatisch ausschaltet
Erfassung der Funktionstasten	Der Zustand (EIN oder AUS) der vier Funktionstasten des Anzeigemoduls kann ausgewertet werden

Tab. 8-8: Funktionen, die durch das SPS-Programm ausgeführt werden können

HINWEIS

Wenn das SPS-Programm durch ein Passwort gegen unbefugtes Lesen oder Schreiben geschützt ist, wird mit dem Anzeigemodul nur das Datum und die Uhrzeit dargestellt. Wenn in diesen Fall eine andere Funktion angewählt wird, blinkt die Anzeige für ca. 5 Sekunden.

8.11.2 Operanden zur Steuerung des Anzeigemoduls

Die Sonderregister D8158 und D8159 dienen zur Steuerung des Anzeigemoduls. In diesen Registern wird die Anfangsadresse eines Register- bzw. eines Merkerbereiches als Konstante eingetragen. Diese beiden Bereiche (5 Register und 15 Merker) enthalten die Einstellungen zur Steuerung des Anzeigemoduls.

Sonderregister		Operand zur Steuerung	Funktion	Beschreibung
Adresse	Inhalt			
D8158	Kn	Dn	Festlegung der angezeigten Operanden	Art der Operanden
		Dn + 1		Adresse des Operanden
		Dn + 2	Automatisches Ausschalten der Beleuchtung der Anzeige	Zeit bis zum Ausschalten der Beleuchtung [Minuten]
		Dn + 3	Schutz vor unbefugte "Zugriffe"	Schutzfunktion
		Dn + 4	Reserviert	
D8159	Kn	Mn	Festlegung der angezeigten Operanden	Freigabe zur Änderung von Operanden
		Mn + 1		Rückmeldung: Änderung abgeschlossen
		Mn + 2	Automatisches Ausschalten der Beleuchtung der Anzeige	Automatisches Ausschalten sperren, Beleuchtung einschalten
		Mn + 3	Einschalten der Fehleranzeige	
		Mn + 4	Abfrage der Funktionstasten des Anzeigemoduls. Solange eine Taste betätigt wird, ist der entsprechende Merker gesetzt.	Status der "ESC"-Taste
		Mn + 5		Status der "-"-Taste
		Mn + 6		Status der "+"-Taste
		Mn + 7		Status der "OK"-Taste
		Mn + 8	Reserviert	
		Mn + 9		
		Mn + 10		
		Mn + 11		
		Mn + 12		
		Mn + 13		
		Mn + 14		

Tab. 8-9: Register- und Merkerbereich zur Steuerung der Anzeige

Wenn in D8158 oder D8159 ein negativer Wert oder ein Wert, der außerhalb des Register- oder Merkerbereiches liegt, eingetragen wird, werden die Steuerfunktionen nicht ausgeführt. Nach dem Einschalten der Steuerung ist der Inhalt der Register D8158 und D8159 "-1", die Steuerfunktionen sind abgeschaltet.

8.11.3 Festlegung der angezeigten Operanden

Durch einen Eintrag in das erste Register des in D8158 angegebenen Registerbereiches können die Operanden festgelegt werden, die mit dem Anzeigemodul dargestellt werden.

Wert in Dn	Ausgewählter Operand
1	Eingänge (X)
2	Ausgänge (Y)
3	Merker (M)
4	Schrittmerker (S)
5	Timer (T)
6	Zähler (C), Soll- und Istwerte von 16-Bit-Zählern und Sollwerte von 32-Bit-Zählern
7	16-Bit-Register (D)
8	32-Bit-Register (D)
9	Datum und Uhrzeit
10	Zähler (C), Soll- und Istwerte von 16-Bit-Zählern und Istwerte von 32-Bit-Zählern

Tab. 8-10:

Auswahl der Operanden durch Eintrag eines Wertes in Dn

HINWEISE

Wenn in dem Register ein Wert eingetragen wird, der außerhalb des Bereiches 1 bis 10 liegt, wird kein Operand ausgewählt und alle Bedienerfunktionen sind erlaubt.

Wenn bei der Festlegung von Timer- und Zähleradressen ein Operand angegeben wird, der nicht im Programm verwendet wird, wird der Operand angezeigt, dessen Adresse dem gewünschten Operanden am nächsten liegt und der im Programm benutzt wird. Liegt die angegebene Adresse außerhalb des zulässigen Bereiches, wird der Operand mit der höchsten Adresse angezeigt, der im Programm verwendet wird. Wenn für den gewählten Timer oder Zähler keine OUT-Anweisung im Programm vorhanden ist, wird "----" angezeigt.

Mit der "+"- und der "-"-Taste am Anzeigemodul können aufeinanderfolgende Operanden ausgewählt werden.

Wenn die Operanden nicht aufeinanderfolgend sind und trotzdem die Auswahl mit der "+"- und der "-"-Taste erfolgen soll, muss das SPS-Programm erweitert werden. Die Operandenadressen, die angezeigt werden sollen, werden dabei in einem Index-Register abgelegt. Eine Betätigung der "+"- und der "-"-Taste wird vom Programm erfasst, der Inhalt des Registers wird erhöht oder erniedrigt und dadurch die Operandenadresse verändert.

Der Inhalt der Registerbereiches wird auf "0" gesetzt, wenn die Spannungsversorgung der Steuerung ausgeschaltet wird und die Register nicht im gepufferten Bereich liegen. Dadurch wird kein Operand mehr ausgewählt und alle Bedienerfunktionen sind erlaubt. Wählen Sie aus diesem Grund für die Steuerfunktionen einen Registerbereich, der auch bei ausgeschalteter Spannung seine Daten nicht verliert.

8.11.4 Schutz vor unbefugte “Zugriffe”

In das vierte Register ($Dn + 3$) des in D8158 angegebenen Registerbereiches wird ein Wert eingetragen, mit dem die Funktionen eingeschränkt werden können.

Wert in $Dn + 3$	Beschreibung
0	Kein Schutz, alle Funktionen sind erlaubt
1	Nur das Datum und die Uhrzeit wird angezeigt, das Datum und die Uhrzeit können nicht verändert werden
2	Nur die Beobachtung von Operanden ist möglich, die Werte der Operanden können nicht verändert werden
Alle anderen Werte	Kein Schutz, alle Funktionen sind erlaubt

Tab. 8-11: Schutzfunktionen

8.11.5 Änderung des Zustandes oder der Werte von Operanden

Diese Funktion ermöglicht dem Bediener die Änderung der angezeigten Operanden mit Hilfe der Funktionstasten.

Sonderregister		Operand zur Steuerung	Beschreibung
Adresse	Inhalt		
D8159	Kn	Mn	Freigabe zur Änderung von Operanden
		Mn + 1	Rückmeldung: Änderung abgeschlossen

Tab. 8-12: Merker zur Änderung von Operanden

Um den Wert eines Operanden zu verändern, muss der Merker Mn gesetzt sein. Bei rückgesetztem Merker Mn ist die Funktion gesperrt. Aus diesem Grund wird die Anwendung der SET-Anweisung empfohlen.

Bei gesetztem Merker Mn können Ausgänge (Y), Merker (M) und Schrittmerker (S) gesetzt oder rückgesetzt werden und die Soll- und Istwerte der Wort-Operanden D, T und C verändert werden.

Um Bit-Operanden zu ändern, wird der blinkende Cursor unter den Operanden plziert. Durch Betätigung der “OK”-Taste wird der Operanden gesetzt oder rückgesetzt. Mit der “ESC”-Taste wird der Editier-Modus verlassen. Der Merker ($Mn + 1$) wird ausgeschaltet und Mn wird rückgesetzt.

Nach Betätigung der “OK”-Taste wird bei Wort-Operanden der aktuelle Wert blinkend dargestellt. Mit der “+”- und der “-”-Taste kann der Wert verändert werden. Nach erneuter Betätigung der “OK”-Taste wird der Wert übernommen.

Wenn die “ESC”-Taste vor der “OK”-Taste betätigt wird, wird die Änderung abgebrochen. Nach der Übernahme des Wertes wird mit der “ESC”-Taste der Editier-Modus verlassen. Der Merker ($Mn + 1$) wird ausgeschaltet und Mn wird rückgesetzt.



Wenn zur Auswahl der Operanden die Betätigung der “+”- und der “-”-Taste vom SPS-Programm erfasst wird, muss eine Verriegelung für den Fall vorgesehen werden, dass mit den beiden Tasten der Zustand oder der Wert von Operanden geändert wird.

8.11.6 Automatisches Ausschalten der Beleuchtung der Anzeige

Die Beleuchtung der Anzeige schaltet sich nach einer bestimmten Zeit automatisch aus. Diese Zeit wird in einem Register vorgegeben. Zusätzlich kann die Beleuchtung ständig ein- oder ausgeschaltet werden.

Sonderregister		Operand zur Steuerung	Funktion	Beschreibung
Adresse	Inhalt			
D8158	Kn	Dn + 2	Zeit bis zum Ausschalten der Beleuchtung [Minuten]	0 (Voreinstellung): 10 Minuten 1 bis 240: 1 bis 240 Minuten >240: 240 Minuten Negativer Wert: Ständig AUS
D8159	Kn	Mn + 2	Automatisches Ausschalten sperren, Beleuchtung einschalten	0: Automatisches Ausschalten freigegeben 1: Automatisches Ausschalten gesperrt, Licht ist immer eingeschaltet

Tab. 8-13: Register und Merker zur Steuerung der Beleuchtung

HINWEIS

Nachdem die Beleuchtung verloschen ist, wird sie wieder eingeschaltet, sobald eine Taste des Anzeigemoduls betätigt wird. Die Anzeige zeigt dasselbe wie vor dem Ausschalten der Beleuchtung. Diese erste Tastenbetätigung dient nur zum Einschalten der Beleuchtung und hat sonst keine Funktion.

8.11.7 Freigeben und Sperren der Fehleranzeige

Durch Setzen oder Rücksetzen des Merkers (Mn + 3) kann ausgewählt werden, welche Fehlermeldungen angezeigt werden.

Sonderregister		Operand zur Steuerung	Funktion	Beschreibung
Adresse	Inhalt			
D8159	Kn	Mn + 3	Einschalten der Fehleranzeige	0: Anzeige von PLC-Hardware-, Parameter-, Programmsyntax- und Programmierfehlern 1: Zusätzliche Anzeige von Konfigurations-, Parallel-Link-Fehlern und Fehlern beim Betrieb

Tab. 8-14: Register- und Merkerbereich zur Steuerung der Anzeige

Bei Betätigung einer beliebigen Taste des Anzeigemoduls oder wenn der Fehler nicht mehr ansteht, verschwindet die Fehleranzeige.

Treten mehrere Fehler gleichzeitig auf, hat die Anzeige der Fehlercodes Vorrang, die unbedingt (auch wenn (Mn + 3) = 0 ist) angezeigt werden. Zusätzlich hat der Fehler mit dem niedrigeren Fehlercode höhere Priorität.

9 Sondermerker, Sonderregister

9.1 Sondermerker (M8000–M8255)

Durch Einsatz von Sondermerkern können Sie bestimmte SPS-Betriebszustände in einem SPS-Programm abfragen bzw. ein- oder ausschalten.

Die Sondermerker lassen sich in zwei Gruppen einteilen:

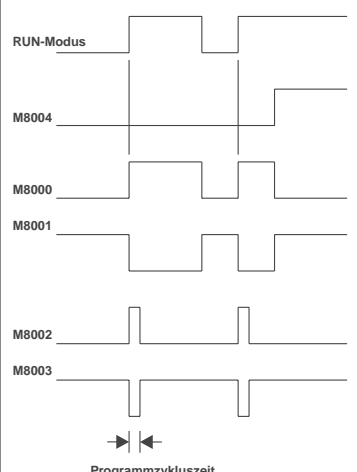
- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontakthanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

HINWEIS

Sondermerker, die Hardware- oder Programmablauf-Fehler anzeigen, sind im folgenden Kapitel 10 beschrieben.

9.1.1 SPS-Status (M8000–M8009)

Sondermerker Operanden-adresse	① Signal-zustand abfragen	② Signal-zustand festlegen	CPU	Bedeutung
M8000	●	—	FX1S FX1N FX2N	SPS-Status anzeigen: RUN
M8001	●	—	FX1S FX1N FX2N	SPS-Status anzeigen: RUN
M8002	●	—	FX1S FX1N FX2N	Initialisierungs-impuls
M8003	●	—	FX1S FX1N FX2N	Initialisierungs-impuls
M8004	●	—	FX1S FX1N FX2N	SPS -Fehler
M8005	●	—	FX1S FX1N FX2N	Der Merker wird gesetzt, wenn die Batteriespannung den in D8006 angegebenen Wert unterschreitet.
M8006	●	—	FX1S FX1N FX2N	Latched den Fehler "niedrige Batteriespannung".
M8007	●	—	FX1S FX1N FX2N	Wird bei kurzzeitigem Spannungsausfall gesetzt.
M8008	●	—	FX1S FX1N FX2N	Meldet einen Spannungsausfall.
M8009	●	—	FX1S FX1N FX2N	Meldet den Ausfall der 24-V-DC Versorgungsspannung.



RUN-Modus

M8004

M8000

M8001

M8002

M8003

→|←
Programmzykluszeit

C000208C

Tab. 9-1: Sondermerker für SPS-Status

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

HINWEIS

Die Beschreibung der Sondermerker M8005 bis M8009 ist den entsprechenden Bedienungsanleitungen (Hardware-Beschreibungen) zu den CPUs zu entnehmen.

9.1.2 Zeittakt (M8011–M8019)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8010	—	—	—	Reserviert
M8011	●	—	FX1S/FX1N FX2N	Taktgeber: 10 ms Es wird ein Takt von 10 ms erzeugt.
M8012	●	—	FX1S/FX1N FX2N	Taktgeber: 100 ms Es wird ein Takt von 100 ms erzeugt.
M8013	●	—	FX1S/FX1N FX2N	Taktgeber: 1 s Es wird ein Takt von 1 s erzeugt.
M8014	●	—	FX1S/FX1N FX2N	Taktgeber: 1 min Es wird ein Takt von 1 min erzeugt.
M8015	●	●	FX1S/FX1N FX2N	Zeiteinstellung Ist der Merker gesetzt, stoppt die Uhr. Die Uhr läuft weiter, wenn der Merker zurück- gesetzt ist.
M8016	●	—	FX1S/FX1N FX2N	Registerdaten Ist der Merker gesetzt, werden die Inhalte von D8013 bis D8019 eingefroren, die Uhr aber läuft weiter.
M8017	●	●	FX1S/FX1N FX2N	Minuten runden Bei gepulstem Signal wird die Echtzeituhr (RTC) auf den vollen Minutenbetrag ge- rundet.
M8018	●	—	FX1S/FX1N FX2N	Echtzeituhr aktiv Sobald der Merker gesetzt ist, wird die Echtzeituhr aktiviert.
M8019	●	—	FX1S/FX1N FX2N	Einstellfehler Die Einstellung der Uhrzeitdaten erfolgte außerhalb des erlaubten Bereiches.

Tab. 9-2: Sondermerker für Zeittakte und Echtzeituhr

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

9.1.3 Flags (M8020–M8029)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8020	●	—	FX1S/FX1N FX2N	Das Zero Flag wird gesetzt, wenn das Ergebnis einer Addition oder Subtraktion gleich Null ist.
M8021	●	—	FX1S/FX1N FX2N	Das Borrow Flag wird gesetzt, wenn das Ergebnis einer Subtraktion (FNC21) kleiner als der kleinste negative Wert ist.
M8022	●	—	FX1S/FX1N FX2N	Das Carry Flag wird gesetzt bei einem Zahlenwertübertrag, bei einer Addition oder bei einem Datenübertrag, bei Ausführung einer Verschiebeanweisung.
M8025	—	—	FX2N	Bei gesetztem Merker werden alle HSC-Anweisungen (FNC 53–55) verarbeitet, sobald der externe HSC-Rücksetzeingang aktiviert wird.
M8026	—	—	FX2N	RAMP-Anweisung (FNC 67) wird gehalten.
M8027	—	—	FX2N	Daten-String für 16 Elemente in der PR-Anweisung (FNC 77).
M8028	●	—	FX1S	Die Benutzung der 10-ms-Timer wird ermöglicht.
M8029	●	—	FX1S/FX1N FX2N	Anweisung vollständig abgearbeitet

Tab. 9-3: Sondermerker für Flags

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

9.1.4 SPS-Modus (M8030–M8039)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8030	●	—	FX2N	Batteriespannung ist zu niedrig, obwohl die BATT. LED nicht leuchtet.
M8031	●	●	FX1S FX1N FX2N	Alle Operanden zurücksetzen, deren Datenwerte nicht im gelatchten Speicher abgespeichert werden.
M8032	●	●	FX1S FX1N FX2N	Alle Operanden zurücksetzen, deren Datenwerte im gelatchten Speicher abgespeichert werden.
M8033	●	●	FX1S FX1N FX2N	Datenwernerhalt im STOP-Modus Der Inhalt des Imageregisters und des Datenspeichers bleibt erhalten, wenn die SPS vom RUN- in den STOP-Modus geschaltet wird.
M8034	●	●	FX1S/FX1N FX2N	Verhindern der Ausgabe: Alle Ausgänge werden gesperrt, das Programm wird aber weiter abgearbeitet
M8035	●	●	FX1S/FX1N FX2N	RUN/STOP-Modus zwangsweise setzen
M8036	●	●	FX1S/FX1N FX2N	Erzwungener RUN-Modus durch Setzen des Sondermerkers im SPS-Programm
M8037	●	●	FX1S/FX1N FX2N	Erzwungener STOP-Modus durch Setzen des Sondermerkers im SPS-Programm
M8038	—	●	FX1S/FX1N FX2N	Einstellung von Geräten für ein n-zu-n-Netzwerk
M8039	●	●	FX1S/FX1N FX2N	SPS-Programm mit konstanter Programmzykluszeit. Wird M8039 gesetzt, arbeitet die SPS das Programm mit einer konstanten Programmzykluszeit ab, dessen Wert im Datenregister D8039 abgespeichert ist.

Tab. 9-4: Sondermerker für SPS-Modus

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

9.1.5 STL-Status (M8040–M8049)

Sondermerker Operanden-adresse	① Signal-zustand abfragen	② Signal-zustand festlegen	CPU	Bedeutung
M8040	●	●	FX1S FX1N FX2N	Weiterschaltung nicht möglich Wird M8040 gesetzt, ist eine Weiterschaltung nicht möglich.
M8041	●	●	FX1S FX1N FX2N	Beginn einer Weiterschaltung Die Weiterschaltung ist während des Automatikbetriebes möglich.
M8042	●	●	FX1S FX1N FX2N	Startimpuls Bei entsprechender Eingangsbedingung wird ein Startimpuls gegeben.
M8043	●	●	FX1S FX1N FX2N	Rückkehr zur Ausgangsposition ausgeführt M8043 wird gesetzt, wenn die Ausgangsposition erreicht ist.
M8044	●	●	FX1S FX1N FX2N	Bedingung zur Rückkehr in die Ausgangsposition M8044 wird gesetzt, wenn die Ausgangsposition erkannt wurde.
M8045	●	●	FX1S FX1N FX2N	Rücksetzen aller Ausgänge ist nicht möglich Wird M8045 gesetzt, ist es nicht möglich, alle Ausgänge zurückzusetzen.
M8046	●	—	FX1S FX1N FX2N	STL-Status ist gesetzt M8046 wird gesetzt, wenn einer der Schrittstatusoperanden S0 bis S899 und M8047 gesetzt ist.
M8047	●	●	FX1S FX1N FX2N	STL-Status anzeigen Wird M8047 gesetzt, wird in den Datenregistern D8040 bis D8047 der Schrittstatus der ersten 8 Schrittstatusoperanden angezeigt.
M8048	●	—	FX2N	Fehlermerker anzeigen Der Merker ist gesetzt, wenn die Überwachung der Fehlermerker (M8049) aktiviert ist und ein Fehlermerker gesetzt wird.
M8049	—	●	FX2N	Anzeige der Fehlermerker ermöglichen Bei gesetztem Merker kann über D8049 die Anzeige der Fehlermerker aktiviert werden.

Tab. 9-5: Sondermerker für STL-Status

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

9.1.6 Interrupt-Programm (M8050–M8059)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8050	●	●	FX1S/FX1N FX2N	Interrupt-Programm I00** wird nicht ausgeführt. ③
M8051	●	●		Interrupt-Programm I10** wird nicht ausgeführt. ③
M8052	●	●		Interrupt-Programm I20** wird nicht ausgeführt. ③
M8053	●	●		Interrupt-Programm I30** wird nicht ausgeführt. ③
M8054	●	●		Interrupt-Programm I40** wird nicht ausgeführt. ③
M8055	●	●		Interrupt-Programm I50** wird nicht ausgeführt. ③
M8056	●	●	FX2N	Interrupt-Programm I6** wird nicht ausgeführt. ③
M8057	●	●		Interrupt-Programm I7** wird nicht ausgeführt. ③
M8058	●	●		Interrupt-Programm I8** wird nicht ausgeführt. ③
M8059	●	●		Schaltet die Interrupts I010 bis I060 aus ②

Tab. 9-6: Sondermerker für Interrupt-Programm

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.
- ③ Wird die EI-Anweisung (FNC 04) im Anwenderprogramm ausgeführt, werden alle Interrupts solange freigegeben, bis einer der aufgeführten Sondermerker gesetzt wird. In diesem Fall wird für jeden gesetzten Sondermerker das zugehörige Interrupt gesperrt, das heißt, es kann nicht aktiviert werden.

9.1.7 Pulse-Catch-Funktion (nur FX2N)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8170	●	—	FX1S/FX1N FX2N	Impuls-Catch X0 ③
M8171	●	—		Impuls-Catch X1 ③
M8172	●	—		Impuls-Catch X2 ③
M8173	●	—		Impuls-Catch X3 ③
M8174	●	—		Impuls-Catch X4 ③
M8175	●	—		Impuls-Catch X5 ③

Tab. 9-7: Sondermerker für Pulse-Catch-Funktion

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.
- ③ Wird ein Impuls-Signal an einem der Eingänge X0 bis X5 erkannt, wird der entsprechende hier aufgeführte Merker gesetzt. Nach dem Rücksetzen des Sondermerkers wird der Merker automatisch wieder mit dem nächsten Impuls-Signal gesetzt. Auf diese Weise können schnelle Eingangsimpulse erkannt und gespeichert werden.

9.1.8 Link- und Sonderfunktionen (M8070–M8198)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8070	●	●	FX1S/FX1N FX2N	Der Merker wird gesetzt, wenn es sich bei der SPS um eine Master-Station in einer parallelen Link-Verbindung handelt.
M8071	●	●		Der Merker wird gesetzt, wenn es sich bei der SPS um eine Slave-Station in einer parallelen Link-Verbindung handelt.
M8072	●	—		Kennung für eine parallele Link-Verbindung der SPS
M8073	●	—		Kennung, wenn M8070 bzw. M8071 in einer parallelen Link-Verbindung falsch gesetzt sind.
M8099	●	●	FX2N	Freier High-Speed-Timer-Betrieb
M8109	●	—	FX2N	Ausgangsaktualisierungsfehler
M8121	●	—	FX1S/FX1N FX2N	RS-Datenübertragung verzögert
M8122	●	●		RS-Datenübertragung-Flag
M8123	●	●		RS-Datenempfang beendet
M8124	●	●		RS-Carrier-Flag-Erkennung
M8126	●	●		RS485-Flag
M8127	●	●		Flag der Handshake-Anforderung (RS485)
M8128	●	—		Anforderungsfehler-Flag (RS485)
M8129	●	●		Flag der Byte-/Wort-Anforderung: Gesetzt=Byte, nicht gesetzt=Wort (RS485)
M8130	●	●	FX2N	Auswahl der Vergleichstabellen, die mit der HSZ-Anweisung verwendet werden
M8131	●	—	FX2N	Kennung zur Beendigung der Verarbeitung des HSZ-Vergleichs
M8132	●	●	FX2N	Festlegung der Bedeutung der PLSY-Anweisung mit den HSZ-Vergleichstabellen
M8133	●	—	FX2N	Kennung für das Verarbeitungsende für den HSZ-Vergleich (bei Verwendung der PLSY-Anweisung)
M8160	●	●	FX2N	XCH-Anweisung als Byte-Tauschfunktion
M8161	●	●	FX1S/FX1N FX2N	Flag für 8-Bit Modus (RS, ASC, ASCI, HEX, CCD)
M8162	●	●		High-Speed Parallellauf-Modus (32-Bit für jede Bewegungsrichtung)
M8167	●	●	FX2N	Hexadezimal-Format für die HKY-Anweisung
M8168	●	●	FX2N	BCD-Format für die SMOV-Anweisung
M8170 – M8175	●	●	FX1S/FX1N FX2N	Impuls-Catch-Flags für die Eingänge X0 bis X5
M8181 – M8186	●	●	FX2N	Alternative zu I010 bis I060 (für den Einsatz alter Programmiergeräte)

Tab. 9-8: Sondermerker für Link- und Sonderfunktionen (1)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung
M8190 – M8197	●	●	FX2N	Konvertierung bestehender Funktionen (MOV, SMOV, RAMP, FMOV) zur Interpretierung als neue Funktion (für den Einsatz alter Programmiergeräte)
M8198	●	●	FX2N	Vertauschzyklus und Zielangabe für BMOV (um mit alten Programmiergeräten File-Register beschreiben zu können)

Tab. 9-9: Sondermerker für Link- und Sonderfunktionen (2)

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

HINWEIS

Die Hinweise zu den Sondermerkern M8060 bis M8069 in Kapitel 11 (Programmfehler) sind ebenfalls zu beachten.

9.1.9 Auf-/Abwärts-Counter (M8200–M8254)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal zustand festlegen	CPU	Bedeutung
M8200	—	●	FX1N FX2N	<p>Wird einer der nebenstehenden Sondermerker gesetzt, wird der zugehörige Counter als abwärtszählender Counter definiert (M8200 = C200, M8201 = C201 usw.).</p> <p>Die Counter sind aufwärtszählend, solange der zugehörige Sondermerker nicht gesetzt ist.</p>
M8201	—	●		
M8203	—	●		
M8204	—	●		
M8205	—	●		
M8206	—	●		
M8207	—	●		
M8208	—	●		
M8209	—	●		
M8210	—	●		
M8211	—	●		
M8212	—	●		
M8213	—	●		
M8214	—	●		
M8215	—	●		
M8216	—	●		
M8217	—	●		
M8218	—	●		
M8219	—	●		
M8220	—	●		
M8221	—	●		
M8222	—	●		
M8223	—	●		
M8224	—	●		
M8225	—	●		
M8226	—	●		
M8227	—	●		
M8228	—	●		
M8229	—	●		
M8230	—	●		
M8231	—	●		
M8232	—	●		
M8233	—	●		
M8234	—	●		

Tab. 9-10: Sondermerker für Auf-/Abwärts-Counter

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

1-Phasen-Counter mit einem Zähleringang (M8235–M8245)

Sondermerker Operanden-adresse	① Signal-zustand abfragen	② Signal-zustand festlegen	CPU	Bedeutung
M8235	●	●	FX1S FX1N FX2N	Wird M8235 gesetzt, arbeitet C235 abwärtszählend.
M8236	●	●		Wird M8236 gesetzt, arbeitet C236 abwärtszählend.
M8237	●	●		Wird M8237 gesetzt, arbeitet C237 abwärtszählend.
M8238	●	●		Wird M8238 gesetzt, arbeitet C238 abwärtszählend.
M8239	●	●		Wird M8239 gesetzt, arbeitet C239 abwärtszählend.
M8240	●	●		Wird M8240 gesetzt, arbeitet C240 abwärtszählend.
M8241	●	●		Wird M8241 gesetzt, arbeitet C241 abwärtszählend.
M8242	●	●		Wird M8242 gesetzt, arbeitet C242 abwärtszählend.
M8243	●	●		Wird M8243 gesetzt, arbeitet C243 abwärtszählend.
M8244	●	●		Wird M8244 gesetzt, arbeitet C244 abwärtszählend.
M8245	●	●		Wird M8245 gesetzt, arbeitet C245 abwärtszählend.

Tab. 9-11: Sondermerker für 1-Phasen-Counter mit einem Zähleringang

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

1-Phasen-Counter mit zwei Zählengängen (M8246–M8250)

Sondermerker Operanden-adresse	① Signal-zustand abfragen	② Signal-zustand festlegen	CPU	Bedeutung
M8246	●	—	FX1S FX1N FX2N	M8246 wird gesetzt, wenn C246 abwärts zählt. M8246 wird zurückgesetzt, wenn C246 aufwärts zählt.
M8247	●	—		M8247 wird gesetzt, wenn C247 abwärts zählt. M8247 wird zurückgesetzt, wenn C247 aufwärts zählt.
M8248	●	—		M8248 wird gesetzt, wenn C248 abwärts zählt. M8248 wird zurückgesetzt, wenn C248 aufwärts zählt.
M8249	●	—		M8249 wird gesetzt, wenn C249 abwärts zählt. M8249 wird zurückgesetzt, wenn C249 aufwärts zählt.
M8250	●	—		M8250 wird gesetzt, wenn C250 abwärts zählt. M8250 wird zurückgesetzt, wenn C250 aufwärts zählt.

Tab. 9-12: Sondermerker für 2-Phasen-Counter mit zwei Zählengängen**A/B-Phasen-Counter mit zwei Zählengängen (M8251–M8255)**

Sondermerker Operanden-adresse	① Signal-zustand abfragen	② Signal-zustand festlegen	CPU	Bedeutung
M8251	●	—	FX1S FX1N FX2N	M8251 wird gesetzt, wenn C251 abwärts zählt. M8251 wird zurückgesetzt, wenn C251 aufwärts zählt.
M8252	●	—		M8252 wird gesetzt, wenn C252 abwärts zählt. M8252 wird zurückgesetzt, wenn C252 aufwärts zählt.
M8253	●	—		M8253 wird gesetzt, wenn C253 abwärts zählt. M8253 wird zurückgesetzt, wenn C253 aufwärts zählt.
M8254	●	—		M8254 wird gesetzt, wenn C254 abwärts zählt. M8254 wird zurückgesetzt, wenn C254 aufwärts zählt.
M8255	●	—		M8255 wird gesetzt, wenn C255 abwärts zählt. M8255 wird zurückgesetzt, wenn C255 aufwärts zählt.

Tab. 9-13: Sondermerker für A/B-Phasen-Counter mit zwei Zählengängen

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontaktanweisung (z. B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.

9.2 Sonderregister (D8000–D8195)

In den Sonderregistern sind Datenwerte über SPS-Betriebszustände gespeichert. Die Datenwerte können vom SPS-Programm gelesen bzw. auch verändert werden.

Die Sonderregister lassen sich in zwei Gruppen einteilen:

- ① Sonderregister, deren Datenwerte von einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte von einem SPS-Programm gelesen und verändert werden können.

9.2.1 SPS-Status (D8000–D8009)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8000	●	●	FX1S FX1N FX2N	Watch-Dog-Timer in Einheiten von 1 ms einstellen. Standardwert 200 ms
D8001	●	—	FX1S FX1N FX2N	Versionsnummer FX/FX0/FX0S/FX0N: 20V _{VV} FX2N: 24V _{VV} (z.B. FX Version 3.30 → 20330)
D8002	●	—	FX1S FX1N FX2N	Speicherkapazität: 0002 → 2k-Schritte 0004 → 4k-Schritte 0008 → 8k-Schritte
D8003	●	—	FX1S FX1N FX2N	Speichertyp: 00 _H →RAM 01 _H →EPROM 02 _H →EEPROM 0A _H →EEPROM (schreibgeschützt) 10 _H →CPU-Speicher
D8004	●	—	FX1S/FX1N FX2N	Fehlermerkeradresse
D8005	—	—	FX1S/FX1N FX2N	Batteriespannung; Zum Beispiel bedeutet der Wert 36 bedeutet 3,6 V
D8006	—	—	FX1S/FX1N FX2N	Speichern der Batteriespannung, an welcher der Fehler "niedrige Batteriespannung" erkannt werden soll.
D8007	—	—	FX1S/FX1N FX2N	Anzahl kurzzeitiger Spannungsausfälle
D8008	—	—	FX1S/FX1N FX2N	Speichern der Verzögerungszeit vom Spannungsausfall bis zum Herunterfahren der CPU (10 ms Standard).
D8009	—	—	FX1S/FX1N FX2N	Speichern der niedrigsten Geräteadresse, die von einem 24-V-DC-Spannungsausfall betroffen ist.

Tab. 9-14: Sonderregister für SPS-Status

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.2 Zeittakt (D8010–D8019)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8010	●	—	FX1S/FX1N FX2N	Aktuelle Programmzykluszeit in Einheiten von 0,1 ms
D8011	●	—	FX1S/FX1N FX2N	Minimale Programmzykluszeit in Einheiten von 0,1 ms
D8012	●	—	FX1S/FX1N FX2N	Maximale Programmzykluszeit in Einheiten von 0,1 ms
D8013	●	●	FX1S FX1N FX2N	Sekunden-Zähler (0–59) der Echtzeituhr
D8014	●	●		Minuten-Zähler (0–59) der Echtzeituhr
D8015	●	●		Stunden-Zähler (0–23) der Echtzeituhr
D8016	●	●		Tages-Zähler (1–31) der Echtzeituhr
D8017	●	●		Monats-Zähler (1–12) der Echtzeituhr
D8018	●	●		Jahres-Zähler (0–99) der Echtzeituhr
D8019	●	●		Wochentagszähler (0–6) der Echtzeituhr

Tab. 9-15: Sonderregister für Zeittakt und Echtzeituhr

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.3 Flags (D8020–D8029)

Sonderregister Operandenadresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8020	●	●	FX1S FX1N FX2N	EingangsfILTER für die Eingänge X0 bis X7 (FX0/FX0S/FX0N) bzw. X0 bis X17 (FX2N) einstellen. Einstellbarer Zeitwert zwischen 0 ms und 15ms in Einheiten von 1 ms (Standardwert 10 ms)
D8022 – D8027	—	—	—	Reserviert
D8028	●	—	FX1S FX1N FX2N	Aktueller Datenwert im Indexregister Z
D8029	●	—		Aktueller Datenwert im Indexregister V

Tab. 9-16: Sonderregister für Flags

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.4 SPS-Modus (D8030 – D8039)

Sonderregister Operandenadresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8030	●	—	FX1S/FX1N	Gelesener Wert vom Potentiometer VR 1 (0 bis 255)
D8030	●	—	FX2N	Dieses Register dient als 3. Speicher-Operand, wenn Z oder V als Zieladressen für die SPD-Anweisung (FNC 56) ausgewählt wurden.
D8031	●	—	FX1S/FX1N	Gelesener Wert vom Potentiometer VR 2 (0 bis 255)
D8032 – D8038	—	—	—	Reserviert
D8039	●	●	FX1S/FX1N FX2N	Konstante Programmzykluszeit in Einheiten von 1 ms einstellen

Tab. 9-18: Sonderregister für SPS-Modus

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.5 STL-Status (D8040–D8049)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8040	●	—	FX1S/FX1N FX2N	Nummer des 1. aktivierten Schrittstatus
D8041	●	—	FX1S/FX1N FX2N	Nummer des 2. aktivierten Schrittstatus
D8042	●	—	FX1S/FX1N FX2N	Nummer des 3. aktivierten Schrittstatus
D8043	●	—	FX1S/FX1N FX2N	Nummer des 4. aktivierten Schrittstatus
D8044	●	—	FX1S/FX1N FX2N	Nummer des 5. aktivierten Schrittstatus
D8045	●	—	FX1S/FX1N FX2N	Nummer des 6. aktivierten Schrittstatus
D8046	●	—	FX1S/FX1N FX2N	Nummer des 7. aktivierten Schrittstatus
D8047	●	—	FX1S/FX1N FX2N	Nummer des 8. aktivierten Schrittstatus
D8049	●	—	FX2N	Letzter Fehlermerker Das Register speichert den letzten aktiven Fehlermerker aus dem Bereich S900 bis S999.

Tab. 9-17: Sonderregister für STL-Status

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.6 Register für Link- und Sonderfunktionen (D8070 – D8099)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8070	●	—	FX1S/FX1N FX2N	Zeit des Watchdog-Timers 500 ms bei Parallel-Link
D8099	●	●	FX2N	Ring-Timer, einstellbar von 0 bis 32767 in 0,1 ms-Schritten

Tab. 9-19: Sonderregister für Link- und Sonderfunktionen

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.7 Sonstige Register (D8102 – D8109)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8102	●	—	FX2N	Speicherkapazität: 0002 → 2k-Schritte 0004 → 4k-Schritte 0008 → 8k-Schritte 0016 → 16k-Schritte
D8109	●	—	FX2N	Operandenadresse, an der der Ausgangsaktualisierungsfehler aufgetreten ist.

Tab. 9-20: Sonstige Sonderregister

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.8 Register für Kommunikationsadapter (232ADP, 485ADP) (D8120 – D8129)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8120	●	●	FX1S/FX1N FX2N	Kommunikations-Format
D8121	●	●		Nummer der lokalen Station (485-Netzwerk)
D8122	●	—		RS, Menge der zu übertragenden Rest-Daten
D8123	●	—		RS, Menge der empfangenen Daten
D8124	●	●		RS, Telegramm-Header (STX(02 _H))
D8125	●	●		232ADP, Telegramm-Ende (ETX(03 _H))
D8127	●	●		RS485, Kopfadresse der angeforderten Station
D8128	●	●		RS485, Datenlänge der angeforderten Daten
D8129	●	●		RS485, Zeiteinstellung des Time-Out- Timers des Netzwerkes

Tab. 9-21: Sonderregister für Kommunikationsadapter

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.9 Ausführungsregister für HSZ- und PLSY-Anweisungen (D8130 – D8143)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8130	●	—	FX2N	Aktueller Vergleichszyklus der HSZ-Anweisung
D8131	●	—	FX2N	Aktueller Vergleichszyklus der HSZ-Anweisung bei aktivierter PLSY-Anweisung
D8132	●	—	FX2N	Ausgabefrequenz für die PLSY-Anweisung
D8134, D8135	●	—	FX2N	Kopie der Werte für die Vergleichsoperation bei Verwendung der HSZ-Anweisung in Verbindung mit der PLSY-Anweisung (32 Bits)
D8136, D8137	●	—	FX2N	Anzahl der mittels PLSY- und PLSR-Anweisung ausgegebenen Impulse (32 Bits)
D8140, D8141	●	—	FX2N	Anzahl der mittels PLSY- und PLSR-Anweisung an Y0 ausgegebenen Impulse (32 Bits)
D8142, D8143	●	—	FX2N	Anzahl der mittels PLSY- und PLSR-Anweisung an Y1 ausgegebenen Impulse (32 Bits)

Tab. 9-22: Sonderregister für HSZ- und PLSY-Anweisungen

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

9.2.10 Index - Register (D8182 – D8195)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8182	●	—	FX1S/FX1N FX2N	Index-Register Z1
D8183	●	—		Index-Register V1
D8184	●	—		Index-Register Z2
D8185	●	—		Index-Register V2
D8186	●	—		Index-Register Z3
D8187	●	—		Index-Register V3
D8188	●	—		Index-Register Z4
D8189	●	—		Index-Register V4
D8190	●	—		Index-Register Z5
D8191	●	—		Index-Register V5
D8192	●	—		Index-Register Z6
D8193	●	—		Index-Register V6
D8194	●	—		Index-Register Z7
D8195	●	—		Index-Register V7

Tab. 9-23: Index-Register

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

10 Programmfehler

10.1 Fehlererkennung

10.1.1 Sondermerker (M8060–M8069)

Sondermerker Operanden- adresse	① Signal- zustand abfragen	② Signal- zustand festlegen	CPU	Bedeutung	(„PROG-E“/ „ERROR“- LED	SPS- Mo- dus
M8060	●	—	FX1N/FX2N	E/A-Konfigurations- fehler	Aus	RUN
M8061	●	—	FX1S/FX1N/FX2N	SPS-Hardwarefehler	Blinkt	STOP
M8062	●	—	FX1N/FX2N	PC/HPP Kommunikations- fehler	Aus	RUN
M8063	●	—	FX1S/FX1N/FX2N	Parallel Kommunikations- fehler	Aus	RUN
M8064	●	●	FX1S/FX1N/FX2N	Parameterfehler	Blinkt	STOP
M8065	●	—	FX1S/FX1N/FX2N	Programmsyntax- fehler	Blinkt	STOP
M8066	●	—	FX1S/FX1N/FX2N	Programmierfehler	Blinkt	STOP
M8067	●	—	FX1S/FX1N/FX2N	Ausführungsfehler im Operanden- bereich (außer Latch-Operanden)	Aus	RUN
M8068	●	●	FX1S/FX1N/FX2N	Ausführungsfehler im Latch-Operanden- bereich	Aus	RUN
M8069	●	●	FX1S/FX1N/FX2N	E/A-Bus-Fehler ③	—	—

Tab. 10-1: Sondermerker für Fehlererkennung

- ① Sondermerker, bei denen nur der Signalzustand in einem SPS-Programm mit einer Kontakthanweisung (z.B. LD- oder LDI-Anweisung) abgefragt werden kann.
- ② Sondermerker, die direkt mit einer Anweisung in einem SPS-Programm gesetzt bzw. zurückgesetzt werden können.
- ③ Nach dem Setzen von M8069 wird eine Kontrolle des E/A-Bus durchgeführt. Wenn hierbei ein Fehler erkannt wird, wird der Fehlercode von 6130 in das Sonderregister D8069 geschrieben und Sondermerker M8061 gesetzt.

10.1.2 Sonderregister (D8060–D8069)

Sonderregister Operanden- adresse	① Daten- werte lesen	② Daten- werte verändern	CPU	Bedeutung
D8060	●	—	FX1N/FX2N	E/A-Adresse des fehlerhaften Grund- oder Erweiterungsgerätes
D8061	●	—	FX1S/FX1N/FX2N	Die Fehlercodenummer des SPS-Hardwarefehlers wird in D8061 gespeichert.
D8062	●	—	FX1N/FX2N	Fehlercode für Kommunikationsfehler zwischen SPS und Programmiergerät (siehe zugehörige Fehlercodetabelle)
D8063	●	—	FX1S/FX1N/FX2N	Fehlercode für Parallel-Link-Fehler (siehe FX-Hardware-Beschreibung)
D8064	●	—	FX1S/FX1N/FX2N	Die Fehlercodenummer des Parameterfehlers wird in D8064 gespeichert.
D8065	●	—	FX1S/FX1N/FX2N	Die Fehlercodenummer des Programmsyntaxfehlers wird in D8065 gespeichert.
D8066	●	—	FX1S/FX1N/FX2N	Die Fehlercodenummer des Programmierfehlers wird in D8066 gespeichert.
D8067	●	—	FX1S/FX1N/FX2N	Die Fehlercodenummer des Ausführungsfehlers wird in D8067 gespeichert.
D8068	●	●	FX1S/FX1N/FX2N	Die Schrittdresse des Ausführungsfehlers wird in D8068 gespeichert.
D8069	●	—	FX1S/FX1N/FX2N	Die Schrittdresse der Fehler M8065 - M8067 wird in D8069 gespeichert.

Tab. 10-2: Sonderregister für Fehlererkennung

- ① Sonderregister, deren Datenwerte in einem SPS-Programm nur gelesen werden können.
- ② Sonderregister, deren Datenwerte in einem SPS-Programm gelesen und verändert werden können.

10.2 Fehlercodes

10.2.1 Fehlercodes (6101–6409)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
SPS-Hardwarefehler	D8061	0000	Kein Fehler	Prüfen Sie die Verbindung zwischen Programmiergerät und Steuerung. Beheben Sie ggf. den Fehler im Schaltkreis.
		6101	RAM-Fehler	
		6102	Schaltkreis fehlerhaft	
		6103	E/A-Fehler (M8069=EIN)	
		6104	Fehler in der 24-V-DC-Versorgungsspannung (M8069=EIN)	
		6105	Watch-Dog-Timer-Fehler	Die Programmzykluszeit ist größer als der in D8000 angegebene Wert.
Kommunikationsfehler zwischen SPS und Programmiergerät	D8062	0000	Kein Fehler	Beheben Sie die Fehlerursache und wiederholen Sie die Übertragung.
		6201	Paritäts-/Überlauf-/Rahmenfehler	
		6202	Kommunikationszeichen fehlerhaft	
		6203	Sum-Check-Fehler bei Datenübertragung	
		6204	Datenformat fehlerhaft	
		6205	Anweisungsfehler	
Fehler in der Kommunikation zweier Paralleladapter FX-40AV/AP	D8063	0000	Kein Fehler	Überprüfen Sie die Spannungsversorgung und Verdrahtung der beiden Paralleladapter. (In 485-Netzwerken können die Fehler nicht über das Netzwerk übertragen werden und müssen vom Master aus überwacht werden.)
		6301	Paritäts-/Überlauf-/Rahmenfehler	
		6302	Zeichenfehler	
		6303	Sum-Check-Fehler	
		6304	Formatfehler	
		6305	Falscher Befehl (485-Netzwerk) Bei Stationsnummer FF war der empfangene Befehl nicht GW (global)	
		6306	Watch-Dog-Timer-Fehler	
		6312	Zeichenfehler im Parallel-Link	
		6313	Summenprüffehler im Parallel-Link	
		6314	Datenformatfehler im Parallel-Link	
Parameterfehler	D8064	0000	Kein Fehler	Stoppen Sie die SPS und korrigieren Sie die falschen Daten.
		6401	Programm-Sum-Check-Fehler	
		6402	Einstellung der Speicherkapazität fehlerhaft	
		6403	Einstellung für Latch-Operanden fehlerhaft	
		6404	Einstellung für Kommentarbereich fehlerhaft	
		6405	Einstellung für File-Register fehlerhaft	
		6409	Andere falsche Parameter	

Tab. 10-3: Fehlercodes (6101–6409)

10.2.2 Fehlercodes (6501–6511)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
Programmsyntaxfehler	D8065	0000	Kein Fehler	Während der Programmierung wird jedesmal die Anweisung kontrolliert. Sollte ein Programmsyntaxfehler auftreten, beseitigen Sie den Fehler im Programmier-Modus.
		6501	Anweisung, Operandensymbol oder Operandenadresse ist falsch programmiert.	
		6502	Keine OUT-T-Anweisung oder OUT-C-Anweisung vor der Programmierung eines entsprechenden Sollwertes.	
		6503	1) Einer OUT-T-Anweisung oder OUT-C-Anweisung folgt keine Sollwertangabe. 2) Die Anzahl der Operanden für eine Applikationsanweisung ist nicht ausreichend.	
		6504	1) Dieselbe Pointer-Markierung ist öfter verwendet worden. 2) Dieselbe Eingangsbedingung für ein Interrupt-Programm oder einen High-Speed-Counter wurde öfter verwendet.	
		6505	Unzulässige Operandenadresse	
		6506	Ungültige Anweisung	
		6507	Ungültige Pointervergabe (P)	
		6508	Ungültige Interrupt-Pointer-Vergabe (I)	
		6509	Andere Fehler	
		6510	Fehlerhafte Nummer der MC-Verschachtelungsebene (N)	
6511	Die Interrupt- und High-Speed-Counter-Eingangsadressen überlappen.			

Tab. 10-4: Fehlercodes (6501-6511)

10.2.3 Fehlercodes (6601–6609)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
Programmierfehler	D8066	0000	Kein Fehler	<p>Ein Programmfehler tritt auf, wenn eine fehlerhafte Kombination von Anweisungen oder eine falsche Beziehung zwischen paarweise zusammengehörigen Anweisungen auftritt.</p> <p>Die erkannten Fehler müssen Sie im Programmier-Modus korrigieren.</p>
		6601	Die LD- oder LDI-Anweisung wurde 9 Mal oder häufiger nacheinander programmiert.	
		6602	1) Keine LD- oder LDI-Anweisung. 2) Die LD-, LDI-, AND-, ANI-Anweisung wurde unzulässig eingesetzt. 3) Folgende Anweisungen sind nicht korrekt verknüpft: STL, RET, MCR, P, I, EI, DI, IRET, FOR, NEXT, FEND, SRET, END. 4) Die MPP-Anweisung fehlt.	
		6603	Die MPS-Anweisung wurde 12 Mal oder häufiger nacheinander programmiert.	
		6604	Die MPS-, MRD- und MPP-Anweisungen wurden unzulässig eingesetzt.	
		6605	1) Die STL-Anweisung wurde 9 Mal oder häufiger nacheinander programmiert. 2) Die MC-, MCR- oder Interrupt-Anweisung wurde innerhalb eines Schrittes programmiert. 3) Die RET-Anweisung wurde außerhalb des Schrittstatus programmiert oder fehlt im Programm.	
		6606	1) Pointer P oder Interrupt-Pointer I fehlen. 2) Die IRET-/ SRET-Anweisung fehlt. 3) Die IRET-/ SRET- und Interrupt-Anweisungen wurden im Hauptprogramm programmiert. 4) STL/RET/MC oder MCR wurden in einem Unterprogramm oder einer Interrupt-Routine programmiert.	
		6607	1) Unzulässige FOR-NEXT-Anweisungen, 6 oder mehr Ebenen. 2) Die folgenden Anweisungen wurden in der FOR-NEXT-Schleife programmiert: STL, RET, MC, MCR, IRET, FEND, SRET, END.	
		6608	1) Unzulässige MC-, MCR-Anweisungen 2) MCR N0 fehlt. 3) Die IRET-/ SRET oder Interrupt-Anweisung wurde zwischen den MC- und MCR-Blöcken programmiert.	
6609	Andere Fehler			

Tab. 10-5: Fehlercodes (6601–6609)

10.2.4 Fehlercodes (6610–6632)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
Stromlaufplanfehler	D8066	6610	Die LD-/LDI-Anweisung wurde mehr als 8 Mal in Schrittfolge programmiert.	Ein Fehler im Stromlaufplan tritt bei einer falschen Kombination von Anweisungen auf. Beheben Sie die Fehler im Programmiermodus.
		6611	Die Anzahl der LD-/LDI-Anweisungen ist kleiner als die der ANB-/ORB-Anweisungen.	
		6612	Die Anzahl der LD-/LDI-Anweisungen ist größer als die der ANB-/ORB-Anweisungen.	
		6613	Die MPS-Anweisung wurde mehr als 12 Mal in Folge programmiert.	
		6614	Die MPS-Anweisung fehlt.	
		6615	Die MPP-Anweisung fehlt.	
		6616	Inkorrekte Verwendung der MPS-, MRD- und MPP-Anweisung. Unter Umständen fehlt die Spulenangabe.	
		6617	Eine der folgenden Anweisungen ist nicht mit der aktiven Bus-Linie verbunden: STL, RET, MCR, Pointer (P), Interrupt (I), EI, DI, SRET, IRET, FOR, NEXT, FEND und END.	
		6618	Die STL-, RET-, MC- oder MCR-Anweisungen wurden innerhalb einer Unter- oder Interrupt-Routine programmiert.	
		6619	Eine ungültige Anweisung wurde innerhalb einer FOR/NEXT-Schleife programmiert: STL, RET, MC, MCR, I, IRET, SRET	
		6620	Die Verschachtelungsgröße für FOR/NEXT-Schleifen wurde überschritten.	
		6621	Ungleiche Anzahl von FOR- und NEXT-Anweisungen.	
		6622	Die NEXT-Anweisung wurde nicht gefunden.	
		6623	Die MC-Anweisung wurde nicht gefunden.	
		6624	Die MCR-Anweisung wurde nicht gefunden.	
		6625	Eine STL-Verzweigungsanweisung treibt mehr als 8 Parallelzweige.	
		6626	Eine ungültige Anweisung wurde innerhalb eines STL-, RET-Blocks programmiert: MC, MCP, I, IRET, SRET.	
		6627	Die RET-Anweisung wurde nicht gefunden.	
		6628	Inkorrekte Programmierung einer I-, IRET- oder SRET-Anweisung im Hauptprogramm	
		6629	Das Pointer (P)- oder Interrupt (I)-Label wurde nicht gefunden.	
6630	Die SRET- oder IRET- Anweisung wurde nicht gefunden.			
6631	Die SRET-Anweisung wurde an einer ungültigen Stelle programmiert.			
6632	Die IRET-Anweisung wurde an einer ungültigen Stelle programmiert.			

Tab. 10-6: Fehlercodes (6610–6632)

10.2.5 Fehlercodes (6701–6709)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
Ausführungsfehler	D8067	0000	Kein Fehler	Diese Fehler treten während der Abarbeitung einer Anweisung auf. Im Fehlerfall müssen Sie die SPS stoppen und die Fehler im Programmier-Modus beseitigen. Ein Ausführungsfehler kann selbst dann auftreten, wenn kein Syntax- oder Programmfehler gemeldet wird. (So ist D500Z z.B. eine gültige Darstellung. Hat Z jedoch einen Wert von 100, wird versucht auf das Datenregister D600 zuzugreifen. In diesem Fall tritt ein Fehler auf, da D600 nicht existiert.)
		6701	1) Für die CJ-Anweisung wurde kein Sprungziel angegeben. 2) Eine Pointermarkierung wird in einem Block programmiert, der erst nach der END-Anweisung abgearbeitet wird. 3) Eine unabhängige Sprungmarke wurde innerhalb einer FOR-NEXT-Schleife oder in einer Unteroutine festgelegt.	
		6702	6 oder mehr CALL-Anweisungen	
		6703	3 oder mehr Interrupt-Ebenen	
		6704	6 oder mehr FOR-NEXT-Ebenen	
		6705	In einer Applikationsanweisung wurde ein falscher Operand eingesetzt.	
		6706	Der Operandenbereich oder Datenbereich, der in einer Applikationsanweisung programmiert wurde, liegt außerhalb des zulässigen Bereichs.	
		6707	Der Zugriff ist auf ein File-Register erfolgt, daß sich außerhalb des zulässigen Adresbereiches befindet.	
		6708	Fehler in Verbindung mit einer FROM-/TO-Anweisung	
		6709	Andere Fehler (z. B. fehlende IRET-Anweisung, unzulässige Beziehung zwischen FOR-NEXT usw.)	

Tab. 10-7: Fehlercodes (6701–6709)

10.2.6 Fehlercodes (6730–6752)

Fehler	Sonderregister	Fehlercode	Bedeutung	Fehler beheben
PID-Ausführungsfehler	D8067	6730	Abtastzyklus T_s ($T_s < 0$ oder > 32767)	Der angegebene Parameterwert liegt außerhalb des zulässigen Bereiches. Der PID-Befehl muss zurückgesetzt werden bevor die Ausführung fortgesetzt wird.
		6732	Filterkoeffizient α ($\alpha < 0$ oder ≥ 101)	
		6733	Proportionalkonstante K_p ($K_p < 0$ oder > 32767)	
		6734	Integrationskonstante T_i ($T_i < 0$ oder > 32767)	
		6735	Differentiationverstärkung K_D ($K_D < 0$ oder ≥ 101)	
		6736	Differentiationskonstante T_D ($T_D < 0$ oder > 32767)	
		6740	Die Abtastzeit T_s ist kleiner als die Programm-Zykluszeit	Abtastzeit wird gleich der Zykluszeit gesetzt.
		6742	Der Istwert von Δ ist zu groß	Die betroffenen Daten werden auf den nächsten Grenzwert zurückgesetzt. Bei allen Fehlercodes (außer 6745) ist das entweder -32768 oder +32767. Die Ausführung wird fortgesetzt aber der PID-Befehl sollte zurückgesetzt werden.
		6743	Die berechnete Abweichung e ist zu groß	
		6744	Das Integrationsergebnis ist zu groß	
		6745	Differentiationswert ist zu groß oder Differenzwert übersteigt den zul. Bereich	
		6746	Differentiationsergebnis ist zu groß	
		6747	Das gesamte PID-Ergebnis ist zu groß	
		6750	Sollwert - Istwert < 150 oder der Regelkreis ist instabil (Die Differenz zwischen Soll- und Istwert schwankt zu stark und zu schnell)	
6751	Der Soll-Wert ist erheblich zu groß			
6752	Große Schwankungen bei der Auto-Tuning-Funktion			

Tab. 10-8: Fehlercodes (6730–6752)

HINWEIS

| Weitere Details zu den Fehlercodes der PID-Anweisung enthält Abs. 7.3.8.

A Technische Daten



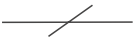


A.1 Übersicht der Grundbefehle

Anweisung	Kontaktplansymbol	Bedeutung	Operanden	Programmschritte	Referenz
LD		LADE; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.2
LDI		LADE NICHT; Beginn einer Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.2
OUT		AUSGABE; Ausgabe, Zuweisung eines Verknüpfungsergebnisses	Y, M, S, T, C	Y, M: 1 S, Sondermerker: 2 T: 3, C: 3 C(32 Bit): 5	Abs. 4.3
AND		UND; UND-Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.4
ANI		UND Nicht; UND-Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.4
OR		ODER; ODER-Verknüpfung mit Abfrage auf Signalzustand „1“	X, Y, M, S, T, C	1	Abs. 4.5
ORI		ODER Nicht; ODER-Verknüpfung mit Abfrage auf Signalzustand „0“	X, Y, M, S, T, C	1	Abs. 4.5
LDP		LADE; (gepulst) Beginn einer Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, G	2	Abs. 4.6
LDF		LADE; (gepulst) Beginn einer Verknüpfung mit Abfrage der abfallenden Flanke	X, Y, M, S, T, G	2	Abs. 4.6
ANP		UND; (gepulst) UND-Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, G	2	Abs. 4.7

Tab. A-1: Grundbefehlsübersicht (Teil 1)

Anweisung	Kontaktplan-symbol	Bedeutung	Operanden	Programmschritte	Referenz
ANF		UND; (gepulst) UND-Verknüpfung mit Abfrage der abfallenden Flanke	X, Y, M, S, T, G	2	Abs. 4.7
ORP		ODER; ODER-Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, G	2	Abs. 4.8
ORF		ODER; ODER-Verknüpfung mit Abfrage der abfallenden Flanke	X, Y, M, S, T, G	2	Abs. 4.8
ANB		UND-Block; Koppelbefehl: Reihenschaltung von Parallelverknüpfungen	—	1	Abs. 4.9
ORB		ODER-Block; Koppelbefehl: Parallelschaltung von Reihenverknüpfungen	—	1	Abs. 4.10
MPS		Push Down Stack; Abspeichern eines Verknüpfungsergebnisses	—	1	Abs. 4.11
MRD		Read Down Stack; Lesen eines Verknüpfungsergebnisses	—	1	Abs. 4.11
MPP		Pop Up Stack; Lesen und Löschen des Verknüpfungsspeichers	—	1	Abs. 4.11
MC		Master Control; Setzen einer Kontrollbedingung	Y, M, keine Sondermerker	3	Abs. 4.12
MCR		Master Control Reset; Rücksetzen einer Kontrollbedingung	N	2	Abs. 4.12
SET		Setzen; Operanden setzen	Y, M, S	Y, M: 1 S, Sondermerker: 2	Abs. 4.13
RST		Rücksetzen; Operanden rücksetzen	Y, M, S, D V, Z, T, C	Y, M: 1 D, V, Z, Sondermerker: 3 T, C: 2	Abs. 4.13

Tab. A-2: Grundbefehlsübersicht (Teil 2)

Anweisung	Kontaktplan-symbol	Bedeutung	Operanden	Programmschritte	Referenz
PLS		Impulserzeugung; Erzeugen eines einmaligen Impulses bei ansteigender Flanke	Y, M	2	Abs. 4.14
PLF		Impulserzeugung; Erzeugen eines einmaligen Impulses bei abfallender Flanke	Y, M	2	Abs. 4.14
INV		Inversion; Umkehrung von Verarbeitungsergebnissen	—	1	Abs. 4.15
NOP		Leerzeile; Leerzeile ohne Funktion	—	1	Abs. 4.16
END		Ende; SPS-Programmende	—	1	Abs. 4.17

Tab. A-3: Grundbefehlsübersicht (Teil 3)

A.2 Allgemeine Systemdaten MELSEC FX1S

Merkmal	Technische Daten
Programmbearbeitung	Zyklische Abarbeitung des gespeicherten Programms
Ein-/Ausgangsbearbeitung	Prozeßabbildverarbeitung Direktverarbeitende Anweisung vorhanden Eingangsfiler von 0 bis 15 ms einstellbar
Programmiersprache	Anweisungsliste und Kontaktplan nach DIN 19239 Schrittanweisungen
Ausführungszeiten der Anweisungen	Grundbefehl: 0,55 bis 0,7 μ s Applikationsanweisungen: siehe Anhang B
Programmkapazität	2k-Schritte, EEPROM-Baustein
Anzahl der Anweisungen	Grundbefehlssatz: 29 Schrittsteueranweisungen: 2 Applikationsanweisungen: 85

Tab. A-4: Allgemeine Systemdaten MELSEC FX1S

A.3 Operanden MELSEC FX1S

Merkmal		Technische Daten			
Ein-/Ausgänge	FX1S-10MR-ES/UL FX1S-10MR-DS FX1S-10MT-DSS	X0 – X5 Y0 – Y3	6 Eingänge 4 Ausgänge		
	FX1S-14MR-ES/UL FX1S-14MR-DS FX1S-14MT-DSS	X0 – X7 Y0 – Y5	8 Eingänge 6 Ausgänge		
	FX1S-20MR-ES/UL FX1S-20MR-DS FX1S-20MT-DSS	X0 – X13 Y0 – Y7	12 Eingänge 8 Ausgänge		
	FX1S-30MR-ES FX1S-30MR-DS FX1S-30MT-DSS	X0 – X17 Y0 – Y15	16 Eingänge 14 Ausgänge		
Merker	Merker	M0 – M383	384 Adressen		
	Latch-Merker	M384 – M511	Istwert im EEPROM gespeichert	128 Adressen	
	Sondermerker	M8000 – M8254	256 Adressen		
Schrittstatus	Initialisierung	S0 – S9	10 Adressen		
	Allgemein	S10 – S127	118 Adressen		
Timer	100 ms	0,1 – 3 276,7s	T0 – T62	63 Adressen	
	10 ms	0,1 – 327,67s	T32 – T62	31 Adressen, wenn M8028 aktiviert	
	1 ms	0,1 – 32,767s	T63	1 Adresse	
Counter	Aufwärtszählend	16 Bit +1 bis +32 767	Allgemein	C0 – C15	16 Adressen
			Istwert im EEPROM gespeichert	C16 – C31	16 Adressen
High-Speed-Counter	1-Phasen-Counter ohne Start und Reset, Auf-/Abwärtszählend	32 Bit	Istwert im EEPROM gespeichert	C235 – C240	6 Zähler
	1-Phasen-Counter mit Start und Reset, Auf-/Abwärtszählend	32 Bit		C241 – C245	5 Zähler
	2-Phasen-Counter, Auf-/Abwärtszählend	32 Bit		C246 – C250	5 Zähler
	A/B-Phasen-Counter	32 Bit		C251 – C255	5 Zähler
Register	Datenregister	16 Bit	Allgemein	D0 – D127	128 Adressen
			Istwert im EEPROM gespeichert	D128 – D255	128 Adressen
	File-Register	Festlegung durch Parameter in 3 Blöcken zu 500 Programmschritten	D1000 – D2499	1500 Adressen	
	Sonderregister	16 Bit	D8000 – D8255	256 Adressen	
	Extern veränderbare Register	16 Bit, Wertebereich von 0 bis 255 durch VR1 und VR2 einstellbar	D8030, D8031	2 Adressen	
	Indexregister	16 Bit	V, Z	16 Adressen	

Tab. A-5: Operanden MELSEC FX1S (1)

Merkmal		Technische Daten		
Pointer	Pointer Sprunganweisung		P0 – P63	64 Adressen
	Interrupt-Pointer	Interrupt-Eingänge: X0 bis X3	I00* – I130*	6 Adressen
Nesting	Programmverzweigung, Hauptkontakt		N0 – N7	8 Adressen
Konstanten	Dezimal	16 Bit	-32 768 – +32 767	
		32 Bit	-2 147 483 648 – +2 147 438 647	
	Hexdezimal	16 Bit	0 – FFFF _H	
		32 Bit	0 – FFFFFFFF _H	

Tab. A-6: Operanden MELSEC FX1S (2)

A.4 Allgemeine Systemdaten MELSEC FX1N

Merkmal	Technische Daten
Programmabarbeitung	Zyklische Abarbeitung des gespeicherten Programms
Ein-/Ausgangsbearbeitung	Prozeßabbildverarbeitung Direkverarbeitende Anweisungen vorhanden Eingangsfiter von 0 bis 15 ms einstellbar
Programmiersprache	Anweisungsliste und Kontaktplan nach DIN 19239 Schrittanweisungen
Ausführungszeiten der Anweisungen	Grundbefehl: 0,55 bis 0,7 μ s Applikationsanweisungen: siehe Anhang B
Programmkapazität	8k-Schritte, EEPROM-Baustein
Anzahl der Anweisungen	Grundbefehlssatz: 29 Schrittsteueranweisungen: 2 Applikationsanweisungen: 89

Tab. A-7: Allgemeine Systemdaten MELSEC FX1N

A.5 Operanden MELSEC FX1N

Merkmal		Technische Daten			
Ein-/Ausgänge	FX1N-□□□-MR-DS FX1N-□□□-MR-ES/UL FX1N-□□□-MT-ESS/UL FX1N-□□□-MT-DSS	Die maximale E/A-Hardwarekonfiguration beträgt 128 Ein-/Ausgangsadressen in Summe. Mit der Software können maximal 128 Ein- und 128 Ausgänge adressiert werden.			
Merker	Merker	M0 – M383		384 Adressen	
	Latch-Merker	M384 – M1535	Istwert im EEPROM gespeichert	1152 Adressen	
	Sondermerker	M8000 – M8255		256 Adressen	
Schrittstatus	Initialisierung	S0 – S9		10 Adressen	
	Allgemein	S10 – S999		990 Adressen	
Timer	100 ms	0,1 – 3 276,7s	T0 – T199	200 Adressen	
	10 ms	0,1 – 327,67s	T200 – T245	46 Adressen	
	1 ms	0,001 – 32.767	T246 – T249	4 Adressen	
	100 ms, remanent	0 – 25,5s	T250 – T255	6 Adressen	
Counter	Aufwärtzählend	16 Bit +1 bis +32 767	Allgemein	C0 – C15	16 Adressen
			Istwert im EEPROM gespeichert	C16 – C199	184 Adressen
	Auf-/Abwärtzählend	32 Bit	Allgemein	C200 – C219	16 Adressen
			Istwert im EEPROM gespeichert	C220 – C234	15 Adressen
High-Speed-Counter	1-Phasen-Counter ohne Start und Reset, Auf-/Abwärtzählend	32 Bit	Istwert im EEPROM gespeichert	C235 – C240	6 Adressen
	1-Phasen-Counter mit Start und Reset, Auf-/Abwärtzählend	32 Bit		C241 – C245	5 Adressen
	2-Phasen-Counter, Auf-/Abwärtzählend	32 Bit		C246 – C250	5 Adressen
	A/B-Phasen-Counter	32 Bit		C251 – C255	5 Adressen
Register	Datenregister	16 Bit	Allgemein	D0 – D127 D1000 – D7999	7128 Adressen
			Istwert im EEPROM gespeichert	D128 – D999	872 Adressen
	File-Register	16 Bit	Festlegung durch Parameter in 14 Blöcken zu 500 Programmschritten	D1000 – D7999	7000 Adressen
	Sonderregister	16 Bit		D8000 – D8255	256 Adressen
	Extern veränderbare Register	16 Bit, Werte von 0 bis 255 durch VR1 und VR2 einstellbar		D8030, D8031	2 Adressen
	Indexregister	16 Bit		V, Z	2 Adressen

Tab. A-8: Operanden MELSEC FX1N (1)

Merkmal		Technische Daten		
Pointer	Pointer Sprunganweisung		P0 – P127	128 Adressen
	Interrupt-Pointer	Interrupt-Eingänge: X0 – X3	I00* – I130*	6 Adressen
Nesting	Programmverzweigung, Hauptkontakt		N0 – N7	8 Adressen
Konstanten	Dezimal	16 Bit	-32 768 – +32 767	
		32 Bit	-2 147 483 648 – +2 147 438 647	
	Hexdezimal	16 Bit	0 – FFFF _H	
		32 Bit	0 – FFFFFFFF _H	

Tab. A-9: Operanden MELSEC FX1N (2)

A.6 Applikationsanweisungen MELSEC FX1S/FX1N

Einteilung	Anweisung	FNC	Bedeutung	Referenz
Programmablaufanweisungen	CJ	00	Sprung innerhalb eines Programms	6.2.1
	CALL	01	Aufruf eines Unterprogramms	6.2.2
	SRET	02	Ende eines Unterprogramms	6.2.3
	IRET	03	Interrupt-Programm abschließen	6.2.4
	EI	04	Interrupt-Programm aktivieren	6.2.4
	DI	05	Interrupt-Programm deaktivieren	6.2.4
	FEND	06	Ende eines Programmbereichs	6.2.5
	WDT	07	Watch-Dog-Timer auffrischen	6.2.6
	FOR	08	Anfang einer Programmwiederholung	6.2.7
	NEXT	09	Ende einer Programmwiederholung	6.2.7
Vergleichs- und Transferanweisungen	CMP	10	Numerische Daten vergleichen	6.3.1
	ZCP	11	Numerische Datenbereiche vergleichen	6.3.2
	MOV	12	Datentransfer	6.3.3
	BMOV	15	Block-Datentransfer	6.3.6
	BCD	18	BCD-Konvertierung	6.3.9
	BIN	19	Binär-Konvertierung	6.3.10
Arithmetische Anweisungen	ADD	20	Addition numerischer Daten	6.4.1
	SUB	21	Subtraktion numerischer Daten	6.4.2
	MUL	22	Multiplikation numerischer Daten	6.4.3
	DIV	23	Division numerischer Daten	6.4.4
	INC	24	Inkrementieren	6.4.5
	DEC	25	Dekrementieren	6.4.6
	WAND	26	Logische UND-Verknüpfung	6.4.7
	WOR	27	Logische ODER-Verknüpfung	6.4.8
	WXOR	28	Logische Exklusiv-ODER-Verknüpfung	6.4.9
Verschiebeanweisung	SFTR	34	Binäre Daten bitweise verschieben, rechts	6.5.5
	SFTL	35	Binäre Daten bitweise verschieben, links	6.5.5
	SFWR	38	Schreiben in einen FIFO-Speicher	6.5.8
	SFRD	39	Schreiben aus einem FIFO-Speicher	6.5.9
Datenoperationen	ZRST	40	Operandenbereiche zurücksetzen	6.6.1
	DECO	41	Daten decodieren	6.6.2
	ENCO	42	Daten codieren	6.6.3

Tab. A-10: Übersicht der Applikationsanweisungen FX1S/FX1N (1)

Einteilung	Anweisung	FNC	Bedeutung	Referenz
High-Speed-Anweisungen	REF	50	Ein- und Ausgänge auffrischen	6.7.1
	MTR	52	Einlesen einer Matrix	6.7.3
	DHSCS	53	Setzen durch High-Speed-Counter	6.7.4
	DHSCR	54	Rücksetzen durch High-Speed-Counter	6.7.4
	SPD	56	Geschwindigkeitserkennung	6.7.6
	PLSY	57	Impulsausgabe einer definierten Anzahl von Impulsen	6.7.7
	PWM	58	Impulsausgabe mit Impulsweitenmodulation	6.7.8
	PLSR	59	Ausgabe einer bestimmten Anzahl von Impulsen	6.7.9
Anwendungsbezogene Anweisungen	IST	60	Schrittstatus initialisieren	6.8.1
	ABSD	62	Absoluter Counter-Vergleich	6.8.3
	INCD	63	Inkrementaler Counter-Vergleich	6.8.4
	ALT	66	Flip-Flop-Funktion	6.8.6
	RAMP	67	Rampenfunktion	6.8.7

Tab. A-11: Übersicht der Applikationsanweisungen FX1S/FX1N (2)

HINWEIS

Die Applikationsanweisungen FNC 70 bis FNC 246 werden im Kapitel 7 beschrieben.

A.7 Allgemeine Systemdaten MELSEC FX2N

Merkmal	Technische Daten
Programmabarbeitung	Zyklische Abarbeitung des gespeicherten Programms
Ein-/Ausgangsbearbeitung	Prozeßabbildverarbeitung Direkverarbeitende Anweisungen vorhanden Eingangsfiler von 0 bis 15 ms einstellbar
Programmiersprache	SPS-Befehlsvorrat nach DIN 19239
Ausführungszeiten der Anweisungen	Grundbefehl: 0,08 µs Applikationsanweisung: siehe Anhang B
Programmkapazität	8k-Schritte: interner RAM-Speicher 16k-Schritte: RAM-, EEPROM-Kassette (optional)
Anzahl der Anweisungen	Grundbefehlssatz: 20 STL-Anweisung: 2 Applikationsanweisung: 125

Tab. A-12: Allgemeine Systemdaten MELSEC FX2N

A.8 Operanden MELSEC FX2N

Merkmal		Technische Daten			
Ein-/Ausgänge	FX2N-□□□-MR-DS FX2N-□□□-MR-ES/UL FX2N-□□□-MT-ESS/UL FX2N-□□□-MT-DSS	Die maximale E/A-Hardwarekonfiguration beträgt 255 Ein-/Ausgangsadressen in Summe. Mit der Software können maximal 255 Ein- und 255 Ausgänge adressiert werden.			
Merker	Merker	M0 – M3071		3072 Adressen	
	Latch-Merker	M500 – M3071		2572 Adressen (anteilig)	
	Sondermerker	M8000 – M8255		256 Adressen	
Schrittstatus	Initialisierung	S0 – S9		10 Adressen (anteilig)	
	Allgemein	S0 – S999		1000 Adressen	
	Latch-Merker	S500 – S999		500 Adressen (anteilig)	
	Fehlermerker	S900 – S999		100 Adressen	
Timer	100 ms	0 – 3 276,7 s	T0 – T199	200 Adressen	
	10 ms	0 – 327,67 s	T200 – T245	46 Adressen	
	1 ms (remanent)	0 – 32.767 s	T246 – T249	4 Adressen	
	100 ms (remanent)	0 – 3276,7 s	T250 – T255	6 Adressen	
Counter	Aufwärtszählend 16 Bit	+1 – +32 767	Allgemein	C0 – C199	200 Adressen
			Istwert im EEPROM gespeichert	C100 – C199	100 Adressen (anteilig)
	Aufwärtszählend 32 Bit	+1 – +214748367	Allgemein	C200 – C234	35 Adressen
			Istwert im EEPROM gespeichert	C219 – C234	15 Adressen (anteilig)
High-Speed-Counter	1-Phasen-Counter	-2147483648 – +2147483647	Istwert im EEPROM gespeichert. Zählfrequenz aller Counter ≤ 20kHz	C235 – C240	6 Adressen
	1-Phasen-Counter mit Start- und Reset-Eingang			C241 – C245	5 Adressen
	2-Phasen-Counter			C246 – C250	5 Adressen
	A/B-Phasen-Counter			C251 – C255	5 Adressen
Register	Datenregister	16 Bit	Allgemein	D0 – D7999	8000 Adressen
			Latch	D200 – D7999	7800 Adressen (anteilig)
	File-Register	16 Bit	Festlegung durch Parameter in 14 Blöcken zu 500 Programmschritten	D1000 – D7999	7000 Adressen
	Sonderregister	16 Bit		D8000 – D8255	256 Adressen
	Indexregister	16 Bit		V0 – V7, Z0 – Z7	16 Adressen

Tab. A-13: Operanden MELSEC FX2N (1)

Merkmal		Technische Daten		
Pointer	Pointer Sprunganweisung		P0 – P63	128 Adressen
	Interrupt-Pointer □ =1 (ansteigende Flanke) □ =0 (abfallende Flanke) **= Zeit in ms	Interrupt-Eingänge: X0 – X3	I00□ – I50□	6 Adressen
		Interrupt-Timer	I6** – I8**	3 Adressen
		Interrupt-Counter	I010 – I060	6 Adressen
Nesting	Programmverzweigung, Hauptkontakt		N0 – N7	8 Adressen
Konstanten	Dezimal	16 Bit	-32 768 – +32 767	
		32 Bit	-2 147 483 648 – +2 147 438 647	
	Hexdezimal	16 Bit	0 bis FFFF _H	
		32 Bit	0 bis FFFFFFFF _H	

Tab. A-14: Operanden MELSEC FX2N (2)

A.9 Applikationsanweisungen MELSEC FX2N

Bezeichnung	Symbol	FNC	Bedeutung	Abschnitt
Programm- ablauf- anweisungen	CJ	00	Sprung innerhalb eines Programms	6.2.1
	CALL	01	Aufruf eines Unterprogramms	6.2.2
	SRET	02	Ende eines Unterprogramms	6.2.3
	IRET	03	Interrupt-Programm abschließen	6.2.4
	EI	04	Interrupt-Programm aktivieren	6.2.4
	DI	05	Interrupt-Programm deaktivieren	6.2.4
	FEND	06	Ende eines Programmbereichs	6.2.5
	WDT	07	Watch-Dog-Timer auffrischen	6.2.6
	FOR	08	Anfang einer Programmwiederholung	6.2.7
	NEXT	09	Ende einer Programmwiederholung	6.2.7
Vergleichs- und Transfer- anweisungen	CMP	10	Numerische Daten vergleichen	6.3.1
	ZCP	11	Numerische Datenbereiche vergleichen	6.3.2
	MOV	12	Datentransfer	6.3.3
	SMOV	13	Shift-Transfer	6.3.4
	CML	14	Kopieren und invertieren	6.3.5
	BMOV	15	Block-Transfer	6.3.6
	FMOV	16	Transfer von gleichen Daten	6.3.7
	XCH	17	Austausch von Daten	6.3.8
	BCD	18	BCD-Konvertierung	6.3.9
	BIN	19	Binär-Konvertierung	6.3.10
Arithmetische Anweisungen	ADD	20	Addition numerischer Daten	6.4.1
	SUB	21	Subtraktion numerischer Daten	6.4.2
	MUL	22	Multiplikation numerischer Daten	6.4.3
	DIV	23	Division numerischer Daten	6.4.4
	INC	24	Inkrementieren	6.4.5
	DEC	25	Dekrementieren	6.4.6
	AND	26	Logische UND-Verknüpfung	6.4.7
	OR	27	Logische ODER-Verknüpfung	6.4.8
	XOR	28	Logische Exklusiv-ODER-Verknüpfung	6.4.9
	NEG	29	Negation von Daten	6.4.10
Verschiebe- anweisungen	ROR	30	Rotation nach rechts	6.5.1
	ROL	31	Rotation nach links	6.5.2
	RCR	32	Rotieren von Bits nach rechts	6.5.3
	RCL	33	Rotieren von Bits nach links	6.5.4
	SFTR	34	Binäre Daten bitweise verschieben, rechts	6.5.5
	SFTL	35	Binäre Daten bitweise verschieben, links	6.5.5
	WSFR	36	Daten wortweise nach rechts verschieben	6.5.6
	WSFL	37	Daten wortweise nach links verschieben	6.5.7
	SFWR	38	Schreiben in einen FIFO-Speicher	6.5.8
	SFRD	39	Lesen aus einem FIFO-Speicher	6.5.9

Tab. A-15: Gesamtübersicht der Applikationsanweisungen FX2N (1)

Einteilung	Anweisung	FNC	Bedeutung	Referenz
Datenoperationen	ZRST	40	Operandenbereiche zurücksetzen	6.6.1
	DECO	41	Daten decodieren	6.6.2
	ENCO	42	Daten codieren	6.6.3
	SUM	43	Ermittlung gesetzter Bits	6.6.4
	BON	44	Überprüfen eines Bits	6.6.5
	MEAN	45	Ermittlung von Durchschnittswerten	6.6.6
	ANS	46	Starten eines Zeitintervalls	6.6.7
	ANR	47	Rücksetzen von Anzeige-Bits	6.6.8
	SQR	48	Ermittlung der Quadratwurzel	6.6.9
	FLT	49	Umwandlung des Zahlenformats	6.6.10
High-Speed-Anweisungen	REF	50	Ein- und Ausgänge auffrischen	6.7.1
	REFF	51	Einstellen der Eingangsfiler	6.7.2
	MTR	52	Einlesen einer Matrix (MTR)	6.7.3
	DHSCS	53	Setzen durch High-Speed-Counter	6.7.4
	DHSCR	54	Rücksetzen durch High-Speed-Counter	6.7.4
	DHSZ	55	Bereichsvergleich	6.7.5
	SPD	56	Geschwindigkeitserkennung	6.7.6
	PLSY	57	Impulsausgabe einer definierte Anzahl von Impulsen	6.7.7
	PWM	58	Impulsausgabe mit Impulsweitenmodulation	6.7.8
	PLSR	59	Ausgabe einer bestimmten Anzahl von Impulsen	6.7.9
Anwendungsbezogene Anweisungen	IST	60	Schrittstatus initialisieren	6.8.1
	SER	61	Suchanweisung	6.8.2
	ABSD	62	Absoluter Counter-Vergleich	6.8.3
	INCD	63	Inkrementaler Counter-Vergleich	6.8.4
	TTMR	64	Teaching-Timer	6.8.5
	STMR	65	Sonder-Timer	6.8.6
	ALT	66	Flip-Flop-Funktion	6.8.7
	RAMP	67	Rampenfunktion	6.8.8
	ROTC	68	Rundtisch-Positionierung	6.8.9
SORT	69	Sortieranweisung	6.8.10	

Tab. A-16: Gesamtübersicht der Applikationsanweisungen FX2N (2)

Hinweis

Die Applikationsanweisungen FNC 70 bis FNC 246 werden im Kapitel 7 beschrieben.

B Ausführungszeiten der Anweisungen

B.1 Ausführungszeiten FX1S-/FX1N-Serie

Eine Beschreibung der Fußnoten befindet sich am Ende des Abschnitts.

B.1.1 Grundbefehle und Schrittstatus-Anweisungen

Anweisung	Bedeutung	Operanden	Programm- schritte	Ausführungszeiten [μ s]	
				Einschalt- zeit	Ausschalt- zeit
LD	Beginn einer Verknüpfung, Abfrage auf „1“-Signal	X, Y, M, S, T, C, Sondermerker	1	0,7	
LDI	Beginn einer Verknüpfung, Abfrage auf „0“-Signal				
AND	UND-Verknüpfung, Abfrage auf „1“-Signal		1	0,65	
ANI	UND-Verknüpfung, Abfrage auf „0“-Signal				
OR	ODER-Verknüpfung, Abfrage auf „1“-Signal				
ORI	ODER-Verknüpfung, Abfrage auf „0“-Signal				
LDP	Lade (gepulst); Beginn einer Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, C	1	11,7	
LDF	Lade (gepulst); Beginn einer Verknüpfung mit Abfrage der abfallenden Flanke				
ANDP	UND (gepulst); UND-Verknüpfung mit Abfrage der ansteigenden Flanke				
ANDF	UND (gepulst); UND-Verknüpfung mit Abfrage der abfallenden Flanke				
ORP	ODER (gepulst); ODER-Verknüpfung mit Abfra- ge der ansteigenden Flanke				
ORF	ODER (gepulst); ODER-Verknüpfung mit Abfra- ge der abfallenden Flanke				
ANB	UND-Block, Reihenschaltung von Parallelverknüpfungen	—	1	0,55	
ORB	ODER-Block, Parallelschaltung von Reihenverknüpfungen				

Tab. B-1: Grundbefehle und Schrittstatus-Anweisungen FX1S/FX1N (1)

Anweisung	Bedeutung	Operanden	Programmschritte	Ausführungszeiten [μs]	
				Einschaltzeit	Ausschaltzeit
MPS	Abspeichern eines Verknüpfungsergebnisses	—	1	0,5	
MRD	Lesen eines Verknüpfungsergebnisses			0,55	
MPP	Lesen und Löschen des Verknüpfungsspeichers			0,5	
INV	Inversion, Umkehrung von Verarbeitungsergebnissen	—	1	0,08	
MC	Setzen einer Kontrollbedingung	N - Y, M	3	8,6	8,0
MCR	Rücksetzen einer Kontrollbedingung	N (Nesting)	2	4,1	
NOP	Leerzeile	—	1	0,45	
END	Programmende		1	450	
STL	Schrittstatus ausführen	S	1	15,8 + 8,2n ^①	
RET	Schrittstatus beenden	—	1	4,8	
OUT	Ausgabe, Zuweisung einer Verknüpfung	Y, M	1	0,7	
		S	2	4,4	
		Sondermerker	2	2,8	
		T-K	3	11,2	10,2
		T-D	3	12,2	11,2
		C-K (16 Bit)	3	8,1	6,9
		C-D (16 Bit)	3	9,5	8,0
		C-K (32 Bit)	5	8,1	6,8
C-D (32 Bit)	5	9,5	8,0		
SET	Setzen, Operanden setzen	Y, M	1	0,85	
		S	2	4,2	2,4
		S bei Einsatz in STL-Stufe ^①		18,6 + 6,8n	2,4
		Sondermerker	2	2,8	
RST	Rücksetzen, Operanden rücksetzen	Y, M	1	0,85	
		S	2	3,8	2,4
		Sondermerker	2	2,8	
		T, C	2	8,7	7,3
		D, V, Z, Sonderregister	3	3,8	1,1
PLS	Impulserzeugung bei ansteigender Flanke	Y, M	2	10,8	
PLF	Impulserzeugung bei abfallender Flanke	Y, M	2	10,8	
P	Pointer adressieren	0 bis 63	1	0,45	
I	Interrupt-Pointer adressieren	□□□□	1	0,45	

Tab. B-2: Grundbefehle und Schrittstatus-Anweisungen FX1S/FX1N (2)

B.1.2 Programmverzweigungsanweisungen

Anweisung		Ausführungszeit [μs]		
Symbol	FNC-Anweisung	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
CJ	FNC 00 /S+	7,1	6,4	—
CALL	FNC 01 /S+	9,3	3,2	—
SRET	FNC 02	8,3	—	—
IRET	FNC 03 ②	8,1	—	—
EI	FNC 04 ②	6,0	—	—
DI	FNC 05 ②	5,3	—	—
FEND	FNC 06 ②	450	—	—
WDT	FNC 07	3,7	2,7	—
FOR	FNC 08 /S+ ②	7,5	—	—
NEXT	FNC 09 ②	4,6	—	—

Tab. B-3: Programmverzweigungsanweisungen FX1S/FX1N

B.1.3 Vergleichs- und Transferanweisungen

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
CMP	FNC 10 /S1+ /S2+ /D+	16	40	2,5	—
		32	41	4,5	—
ZCP	FNC 11 /S1+ /S2+ /S+ /D+	16	45	2,5	—
		32	47	4,5	—
MOV	FNC 12 /S+ /D+	16	19	2,5	—
		32	22	3,0	—
BMOV	FNC 15 /S+ /D+ /n ④	16	78 + 22n	2,5	—
BCD	FNC 18 /S+ /D+	16	30	2,5	—
		32	38,6	3,0	—
BIN	FNC 19 /S+ /D+	16	30	2,5	—
		32	35,5	3,0	—

Tab. B-4: Vergleichs- und Transferanweisungen FX1S/FX1N

B.1.4 Arithmetische Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ADD	FNC 20 /S1+ /S2+ /D+ Z, Cy, Br	16	37,5	2,5	—
		32	40,2	4,5	224
SUB	FNC 21 /S1+ /S2+ /D+ Z, Cy, Br	16	37,5	2,5	—
		32	40,5	4,5	232
MUL	FNC 22 /S1+ /S2+ /D+	16	38,2	2,5	—
		32	50,3	4,5	162
DIV	FNC 23 /S1+ /S2+ /D+	16	39,2	2,5	—
		32	63,5	4,5	197
INC	FNC 24 /D+ ③	16	14,5	2,5	—
		32	16,7	4,5	—
DEC	FNC 25 /D+ ③	16	14,5	2,5	—
		32	16,7	4,5	—
WAND	FNC 26 /S1+ /S2+ /D+	16	35,7	2,5	—
		32	37,3	4,5	—
WOR	FNC 27 /S1+ /S2+ /D+	16	35,7	2,5	—
		32	37,3	4,5	—
WXOR	FNC 28 /S1+ /S2+ /D+	16	35,7	2,5	—
		32	37,3	4,5	—

Tab. B-5: Arithmetische Anweisungen FX1S/FX1N

Br (Borrow): M8021
 Cy (Carry): M8022
 F (Anweisung vollständig abgearbeitet): M8029

B.1.5 Rotations- und Shift-Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
SFTR	FNC 34 /S+ /D+ /n1 /n2 ③ ⑥	16	55 + 1,25n	2,5	—
SFTL	FNC 35 /S+ /D+ /n1 /n2 ③ ⑥	16	56,1 + 1,25n	2,5	—
SFWR	FNC 38 /S+ /D+ /n ③ ⑦	16	41,6	2,5	—
SFRD	FNC 39 /S+ /D+ /n ③ ⑦	16	52,3	2,5	—

Tab. B-6: Rotations- und Shift-Anweisungen FX1S/FX1N

B.1.6 Datenoperationen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ZRST	FNC 40 /D1+ /D2+ ⑧	16 (D)	32,4 + 0,5n	2,5	—
		16 (S)	37,8 + 0,9n		
		16 (C)			
		16 (T)			
		16 (M)	51,8 + 0,8n		
		16 (Y)			
DECO	FNC 41 /S+ /D+ /n	16	65,6	2,5	—
ENCO	FNC 42 /S+ /D+ /n	16	46,7	2,5	—

Tab. B-7: Datenoperationen FX1S/FX1N

B.1.7 High-Speed-Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
REF	FNC 50 /D /n ⑩	16	19,5 + 4,3n	2,5	—
MTR	FNC 52 /S /D1 /D2 /n	16	22,6	9,8	—
HSCS	FNC 53 /S1+ /S2+ /D+ ⑫	32	46,8	4,5	—
HSCR	FNC 54 /S1+ /S2+ /D+ ⑫	32	46,8	4,5	—
SPD	FNC 56 /S1+ /S2+ /D+	②	39,5	43,8	—
		16	82,6	22,8	—
PLSY	FNC 57 /S1+ /S2+ /D+	32	100,6	34,9	—
		16	38,7	42,6	—
PWM	FNC 58 /S1+ /S2+ /D+	16	38,7	42,6	—
PLSR	FNC 59 /S1+ /S2+ /S3+ /D+	16	91,6	27,8	—
		32	113,7	41,6	

Tab. B-8: High-Speed-Anweisungen FX1S/FX1N

B.1.8 Anwendungsbezogene Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
IST	FNC 60 /S /D1 /D2	16	81,7	2,5	—
ABSD	FNC 62 /S1+ /S2+ /D+ /n ¹⁴	16	56,5 + 6,3n	2,5	—
		32	62,7 + 11n	2,5	
INCD	FNC 63 /S1+ /S2+ /D+ /n	16	60,5	52,7	—
ALT	FNC 66 /D+	16	21,8	2,5	—
RAMP	FNC 67 /S1+ /S2+ /D+ /n	16	52,5	44,8	—

Tab. B-9: Anwendungsbezogene Anweisungen FX1S/FX1N

B.1.9 Spezielle FNC-Anweisungen

Anweisung		Ausführungszeit [µs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
DSW	FNC 72 /S+ /D1+ /D2+ /n	16	95,0	92,6	—
SEGL	FNC 74 /S+ /D+ /n	16 1Set	84,5	40,7	—
FROM (nur FX1N)	FNC 78 /n1 /n2 /D+ /n3 ⑩	16	87 + 483n	2,5	—
		32	102 + 973n	4,5	
TO (nur FX1N)	FNC 79 /n1 /n2 /S+ /n3 ⑩	16	85 + 542n	2,5	—
		32	98 + 1121n	4,5	
RS	FNC 80	16	56,3	9,2	—
PRUN	FNC 81 /S+ /D+ ⑪	16	46,7 + 1,0n	2,5	—
		32	47,7 + 1,0n	3,0	—
ASCI	FNC 82	16	52,8 + 5,8n	2,5	—
HEX	FNC 83	16	54 + 8,9n	2,5	—
CCD	FNC 84	16	54,3 + 4,5n	2,5	—
VRRD	FNC 85 /S+ /D+	16	142,7	8,9	—
VRSC	FNC 86 /S+ /D+	16	142,7	8,9	—
PID	FNC 88	16	65,5	8,5	—

Tab. B-10: Spezielle FNC-Anweisungen FX1S/FX1N

B.1.10 Positionier-Anweisungen

Anweisung		Ausführungszeiten [µs]		
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit
ABS	FNC 155 /S+ /D1+ /D2+	32	86,7	85,7
ZRN	FNC 156 /S1+ /S2+ /S3+ /D	16	107,8	27,8
		32	130,5	40,8
PLSV	FNC 157 /S+ /D1 /D2+	16	79,6	22,7
		32	97,8	33,5
DRVI	FNC 158 /S1+ /S2+ /D1 /D2+	16	87,7	26,8
		32	110,6	40,7
DRVA	FNC 158 /S1+ /S2+ /D1 /D2+	16	89,6	26,8
		32	112,7	40,7

Tab. B-11: Positionier-Anweisungen FX1S/FX1N

B.1.11 Anweisungen für die Echtzeituhr

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
TCMP	FNC 160 /S1+ /S2+ /S3+ /S+ /D+	16	52,6	2,5	—
TZCP	FNC 161 /S1+ /S2+ /S+ /D+	16	64,7	2,5	—
TADD	FNC 162 /S1+ /S2+ /D+	16	42,9	2,5	—
TSUB	FNC 163 /S1+ /S2+ /D+	16	42,9	2,5	—
TRD	FNC 166 /D+	16	29,7	2,5	—
TWR	FNC 167 /S+	16	633,5	2,5	—
HOUR	FNC 169 /S1+ /D1+ /D2+	16	39,7	38,7	
		32	41,9	40,6	

Tab. B-12: Anweisungen für die Echtzeituhr

B.1.12 Datenaustausch mit Analogeingangsmodule FX0N-3A

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
RD3A (nur FX1N)	FNC 176 /M1+ /M2+ /D+	16	1248,3	7,5	—
WR3A (nur FX1N)	FNC 177 /M1+ /M2+ /D+	16	1263,7	7,5	—

Tab. B-13: Anweisungen der FX1N zur Kommunikation mit FX0N-3A

B.1.13 Vergleichsanweisungen

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
LD□	FNC 221 – 230 /S1+ /S2+	16	27,6		—
		32	28,2		
AND□	FNC 232 – 238 /S1+ /S2+	16	27,6		—
		32	28,2		
OR□	FNC 240 – 246 /S1+ /S2+	16	27,6		—
		32	28,2		

Tab. B-14: Vergleichsanweisungen

- ① „n“ gibt die Anzahl der statischen STL-Anweisungen (Anzahl der parallelen/zusammenführenden Anweisungen) an.
- ② Die gekennzeichneten Anweisungen benötigen keine Kontakte.
- ③ Wird die statische Anweisung eingesetzt und nicht die Pulsanweisung, verändert sich der Wert der Zieladresse.
- ④ „n“ gibt die Anzahl der zu verschiebenden Register an ($n \leq 512$).
- ⑥ „n“ gibt die Anzahl der zu verarbeitenden Bit-Operanden an.
- ⑦ „n“ gibt die Anzahl der zu verarbeitenden Operanden an ($2 \leq n \leq 512$).
- ⑧ „n“ gibt den zurückzusetzenden Operandenbereich an.
Der Operandentyp ist in den Klammern angegeben.
- ⑩ „n“ gibt den zu aktualisierenden Operandenbereich an ($8 \leq n \leq 128$ in 8-ter-Schritten).
- ⑫ Diese Anweisung kann max. 6 mal zur gleichen Zeit aktiv sein.
- ⑭ „n“ gibt die Anzahl der Ausgangsadressen an ($n \leq 64$).
- ⑯ „n“ gibt die Anzahl der in das/aus dem Sondermodul zu schreibenden oder lesenden Datenworte an.
- ⑰ „n“ gibt die Anzahl der zu schreibenden oder lesenden Single-Byte-Datenworte (8 Bit) bei Parallelbetrieb zweier FX-Steuerungen an.

B.2 Ausführungszeiten FX2N

Eine Beschreibung der Fußnoten befindet sich am Ende des Abschnitts.

B.2.1 Grundbefehle und Schrittstatus-Anweisungen

Anweisung	Bedeutung	Operanden	Programmschritte	Ausführungszeiten [µs]	
				Einschaltzeit	Ausschaltzeit
LD	Beginn einer Verknüpfung, Abfrage auf „1“-Signal	X, Y, M, S, T, C, Sondermerker	1	0,08	
LDI	Beginn einer Verknüpfung, Abfrage auf „0“-Signal				
AND	UND-Verknüpfung, Abfrage auf „1“-Signal				
ANI	UND-Verknüpfung, Abfrage auf „0“-Signal				
OR	ODER-Verknüpfung, Abfrage auf „1“-Signal				
ORI	ODER-Verknüpfung, Abfrage auf „0“-Signal				
LDP	Lade (gepulst); Beginn einer Verknüpfung mit Abfrage der ansteigenden Flanke	X, Y, M, S, T, C	1	43,2	
LDF	Lade (gepulst); Beginn einer Verknüpfung mit Abfrage der abfallenden Flanke				
ANDP	UND (gepulst); UND-Verknüpfung mit Abfrage der ansteigenden Flanke				
ANDF	UND (gepulst); UND-Verknüpfung mit Abfrage der abfallenden Flanke				
ORP	ODER (gepulst); ODER-Verknüpfung mit Abfrage der ansteigenden Flanke				
ORF	ODER (gepulst); ODER-Verknüpfung mit Abfrage der abfallenden Flanke				
ANB	UND-Block, Reihenschaltung von Parallelverknüpfungen	—	1	0,08	
ORB	ODER-Block, Parallelschaltung von Reihenverknüpfungen				
MPS	Abspeichern eines Verknüpfungsergebnisses				
MRD	Lesen eines Verknüpfungsergebnisses				
MPP	Lesen und Löschen des Verknüpfungsspeichers				

Tab. B-15: Grundbefehle und Schrittstatus-Anweisungen FX2N (1)

Anweisung	Bedeutung	Operanden	Programmschritte	Ausführungszeiten [μ s]	
INV	Inversion, Umkehrung von Verarbeitungsergebnissen	—	1	0,08	
MC	Setzen einer Kontrollbedingung	N - Y, M	3	24,8	27,5
MCR	Rücksetzen einer Kontrollbedingung	N (Nesting)	2	20,8	
NOP	Leerzeile	—	1	0,08	
END	Programmende		1	508	
STL	Schrittstatus ausführen	S	1	27,3 + 12,6n ^①	
RET	Schrittstatus beenden	—	1	21,6	
OUT	Ausgabe, Zuweisung einer Verknüpfung	Y, M	1	0,08	
		S	2	24,4	24,3
		Sondermerker	2	0,16	0,16
		T-K	3	42,3	37,4
		T-D	3	42,2	37,2
		C-K (16 Bit)	3	25,5	24,9
		C-D (16 Bit)	3	25,3	25,0
		C-K (32 Bit)	5	25,3	24,9
C-D (32 Bit)	5	25,2	24,9		
SET	Setzen, Operanden setzen	Y, M	1	0,08	
		S	2	23,7	17,2
		S bei Einsatz in STL-Stufe ^①		27,3 + 12,6n	
		Sondermerker	2	0,16	0,16
RST	Rücksetzen, Operanden rücksetzen	Y, M	1	0,16	
		S	2	23,1	17,3
		Sondermerker	2	0,16	0,16
		T, C	2	27	25
		D, V, Z, Sonderregister	3	21,9	17,1
PLS	Impulserzeugung bei ansteigender Flanke	Y, M	2	0,32	0,32
PLF	Impulserzeugung bei abfallender Flanke	Y, M	2	0,32	
P	Pointer adressieren	0 bis 63	1	0,08	
I	Interrupt-Pointer adressieren	I□□□	1	0,08	

Tab. B-16: Grundbefehle und Schrittstatus-Anweisungen FX2N (2)

B.2.2 Programmverzweigungsanweisungen

Anweisung			Ausführungszeit [μ s]		
Symbol	FNC-Anweisung		EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
CJ	FNC 00 /S+		29	6,4	—
CALL	FNC 01 /S+		32,2	6,4	—
SRET	FNC 02	②	21,2	21,2	—
IRET	FNC 03	②	18,8	18,1	—
EI	FNC 04	②	55,8	55,8	—
DI	FNC 05	②	18,5	18,5	—
FEND	FNC 06	②	508		—
WDT	FNC 07		26,3	6,4	—
FOR	FNC 08 /S+	②	27,6	27,6	—
NEXT	FNC 09	②	5,2	5,2	—

Tab. B-17: Programmverzweigungsanweisungen FX2N

B.2.3 Vergleichs- und Transferanweisungen

Anweisung			Ausführungszeit [μ s]			
Symbol	FNC-Anweisung		Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
CMP	FNC 10 /S1+ /S2+ /D+		16	87,6	6,4	—
			32	91,9	6,4	—
ZCP	FNC 11 /S1+ /S2+ /S+ /D+		16	103,2	6,4	—
			32	108,9	6,4	—
MOV	FNC 12 /S+ /D+		16	1,52	1,52	—
			32	1,84	1,84	—
SMOV	FNC 13 /S+ /m1 /m2 /D+ /n		16	155,2	6,4	—
CML	FNC 14 /S+ /D+		16	51,4	6,4	—
			32	55,9	6,4	—
BMOV	FNC 15 /S+ /D+ /n	④	16	$97 + 1,7n$	6,4	—
FMOV	FNC 16 /S+ /D+ /n	④	16	$69,1 + 2,8n$	6,4	—
			32	$73,2 + 5,2n$	6,4	—
XCH	FNC 17 /D1+ /D2+	③	16	57,2	6,4	—
			32	64	6,4	—
BCD	FNC 18 /S+ /D+		16	37,9	6,4	—
			32	57,6	6,4	—
BIN	FNC 19 /S+ /D+		16	32,4	6,4	—
			32	44,5	6,4	—

Tab. B-18: Vergleichs- und Transferanweisungen FX2N

B.2.4 Arithmetische Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ADD	FNC 20 /S1+ /S2+ /D+ Z, Cy, Br	16	27,6	6,4	—
		32	28,9	6,4	224
SUB	FNC 21 /S1+ /S2+ /D+ Z, Cy, Br	16	27,6	6,4	—
		32	28,9	6,4	232
MUL	FNC 22 /S1+ /S2+ /D+	16	25,2	6,4	—
		32	31,4	6,4	162
DIV	FNC 23 /S1+ /S2+ /D+	16	32	6,4	—
		32	36,4	6,4	197
INC	FNC 24 /D+ ③	16	18,8	6,4	—
		32	20,2	6,4	—
DEC	FNC 25 /D+ ③	16	18,9	6,4	—
		32	20	6,4	—
WAND	FNC 26 /S1+ /S2+ /D+	16	23,4	6,4	—
		32	24,8	6,4	—
WOR	FNC 27 /S1+ /S2+ /D+	16	23,5	6,4	—
		32	24,7	6,4	—
WXOR	FNC 28 /S1+ /S2+ /D+	16	23,5	6,4	—
		32	25,0	6,4	—
NEG	FNC 29 /D+ ③	16	35,3	6,4	—
		32	38,4	6,4	—

Tab. B-19: Arithmetische Anweisungen FX2N

Br (Borrow): M8021
 Cy (Carry): M8022
 F (Anweisung vollständig abgearbeitet): M8029

B.2.5 Rotations- und Shift-Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ROR	FNC 30 /D+ /n Cy 3 5	16	61,7	6,4	—
		32	65,3	6,4	—
ROL	FNC 31 /D+ /n Cy 3 5	16	61,2	6,4	—
		32	65,2	6,4	—
RCR	FNC 32 /D+ /n Cy 3 5	16	66,3 + 2,2n	6,4	—
		32	69,7 + 2,6n	6,4	—
RCL	FNC 33 /D+ /n Cy 3 5	16	65,8 + 2,2n	6,4	—
		32	69,5 + 2,6n	6,4	—
SFTR	FNC 34 /S+ /D+ /n1 /n2 3 6	16	107 + 53,8n	6,4	—
SFTL	FNC 35 /S+ /D+ /n1 /n2 3 6	16	105 + 53,8n	6,4	—
WSFR	FNC 36 /S+ /D+ /n1 /n2 3 4	16	126 + 11,7n	6,4	—
WSFL	FNC 37 /S+ /D+ /n1 /n2 3 4	16	125 + 11,8n	6,4	—
SFWR	FNC 38 /S+ /D+ /n 3 7	16	83,9	6,4	—
SFRD	FNC 39 /S+ /D+ /n 3 7	16	80,2	6,4	—

Tab. B-20: Rotations- und Shift-Anweisungen FX2N

B.2.6 Datenoperationen

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ZRST	FNC 40 /D1+ /D2+ ⑧	16 (D)	77 + 1,7n	6,4	—
		16 (S)	83 + 11,1n		
		16 (C)			
		16 (T)			
		16 (M)	89,2 + 9,4n		
		16 (Y)			
DECO	FNC 41 /S+ /D+ /n	16	76	6,4	—
ENCO	FNC 42 /S+ /D+ /n	16	81,8	6,4	—
SUM	FNC 43 /S+ /D+	16	72,8	6,4	—
		32	94,6	6,4	—
BON	FNC 44 /S+ /D+ /n	16	78,2	6,4	—
		32	82,3	6,4	—
MEAN	FNC 45 /S+ /D+ /n ⑨	16	83,8 + 3,4n	6,4	—
		32	90,9 + 6,7n	6,4	—
ANS	FNC 46 /S+ /m /D+	16	100,8	6,4	—
ANR	FNC 47 ⑩	16	37,7	6,4	—
SQR	FNC 48	16	150,2	6,4	—
		32	154,8	6,4	344
FLT	FNC 49	16	66,8	6,4	—
		32	66,8	6,4	—

Tab. B-21: Datenoperationen FX2N

B.2.7 High-Speed-Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
REF	FNC 50 /D /n ⑩	16	99,6 + 0,6n	6,4	—
REFF	FNC 51 /n ⑪	16	65,3 + 1,7n	6,4	—
MTR	FNC 52 /S /D1 /D2 /n	16	39,1	23,6	—
HSCS	FNC 53 /S1+ /S2+ /D+ ⑫	32	87,8	6,4	—
HSCR	FNC 54 /S1+ /S2+ /D+ ⑫	32	88,6	6,4	—
HSZ	FNC 55 /S1+ /S2+ /S+ /D+ ⑫	32	100,6	6,4	—
SPD	FNC 56 /S1+ /S2+ /D+	②	80,2	80,2	—
PLSY	FNC 57 /S1+ /S2+ /D+	16	85	73,3	—
		32	86,6	75,8	—
PWM	FNC 58 /S1+ /S2+ /D+	16	70,4	73,3	—
PLSR	FNC 59 /S1+ /S2+ /S3+ /D+	16	122,6	87,5	—
		32	125,6	90,5	

Tab. B-22: High-Speed-Anweisungen FX2N

B.2.8 Anwendungsbezogene Anweisungen

Anweisung		Ausführungszeit [μ s]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
IST	FNC 60 /S /D1 /D2	16	114,3	6,4	—
SER	FNC 61 ⑬	16	129,2 + 8,6n	22,9	—
		32	147 + 9n	29	
ABSD	FNC 62 /S1+ /S2+ /D+/ n ⑭	16	91,8 + 20,2n	6,4	—
		32	97,5 + 21,5n	6,4	
INCD	FNC 63 /S1+ /S2+ /D+/ n	16	110,5	19,5	—
TTMR	FNC 64 /D+ /n	16	54,9	44,9	—
STMR	FNC 65 /S+ /m /D+	16	84,4	84,4	—
ALT	FNC 66 /D+	16	50,1	6,4	—
RAMP	FNC 67 /S1+ /S2+ /D+ /n	16	98,1	81,6	—
ROTC	FNC 68 /S+ /m1 /m2 /D+	16	118,4	107,2	—
SORT	FNC 69 ⑮	16	50,5	19,5	—

Tab. B-23: Anwendungsbezogene Anweisungen FX2N

B.2.9 Spezielle FNC-Anweisungen

Anweisung		Ausführungszeit [μs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
TKY	FNC 70 /S+ /D1+ /D2+	16	97,2	22,2	—
		32	98,7	22,2	—
HKY	FNC 71 /S+ /D1+ /D2+ /D3+	16	92,2	27,4	—
		32	65,0	6,4	—
DSW	FNC 72 /S+ /D1+ /D2+ /n	16	92,2	27,4	—
SEGD	FNC 73 /S1+ /D+	16	65	6,4	—
SEGL	FNC 74 /S+ /D+ /n	16 1Set	105,9	26,5	—
ARWS	FNC 75 /S1+ /D+ /D2+ /n	16	134,4	22,1	—
ASC	FNC 76 /S+ /D+	16	49,5	6,4	—
PR	FNC 77 /S+ /D+	16	während des Druckvorgangs: 114,8 bei beendetem Druckvorgang: 88	88,5	—
FROM	FNC 78 /n1 /n2 /D+ /n3 ¹⁶	16	97 + 487n	6,4	—
		32	99 + 962n	6,4	
TO	FNC 79 /n1 /n2 /S+ /n3 ¹⁶	16	94 + 557n	6,4	—
		32	96 + 1099n	6,4	
RS	FNC 80	16	117,6	18	—
		32			
PRUN	FNC 81 /S+ /D+ ¹⁷	16	65,6 + 17n	6,4	—
		32	67 + 17,7n	6,4	—
ASCI	FNC 82	16	88,2 + 10,8n	6,4	—
HEX	FNC 83	16	89,7 + 20n	6,4	—
CCD	FNC 84	16	90,5 + 4,8n	6,4	—
VRRD	FNC 85 /S+ /D+	16	209,7	27,3	—
VRSC	FNC 86 /S+ /D+	16	202,4	27,3	—
PID	FNC 88	16	155	89	—

Tab. B-24: Spezielle FNC-Anweisungen FX2N (1)

Anweisung		Ausführungszeit [µs]			
Symbol	FNC-Anweisung	Bit	EIN-Schaltzeit	AUS-Schaltzeit	Sonderfunktion der FNC
ECMP	FNC 110 /S1+ /S2+ /D+	32	104,4	6,4	—
EZCP	FNC 111 /S1+ /S2+ /S3+ /D+	32	124,5	6,4	—
EBCD	FNC 118 /S+ /D+	32	106,9	6,4	—
EBIN	FNC 119 /S+ /D+	32	81,3	6,4	—
EADD	FNC 120 /S1+ /S2+ /D+	32	117,4	6,4	—
ESUB	FNC 121 /S1+ /S2+ /D+	32	117,4	6,4	—
EMUL	FNC 122 /S1+ /S2+ /D+	32	96,4	6,4	—
EDIV	FNC 123 /S1+ /S2+ /D+	32	100,4	6,4	—
ESQR	FNC 127 /S+ /D+	32	152,1	6,4	—
INT	FNC 129 /S+ /D+	16	67,5	6,4	—
		32	70,4	6,4	
SIN	FNC 130 /S+ /D+	32	199,5	6,4	—
COS	FNC 131 /S+ /D+	32	262,5	6,4	—
TAN	FNC 132 /S+ /D+	32	425,3	6,4	—
SWAP	FNC 147 /S+	16	36,1	6,4	—
		32	41,2	6,4	
TCMP	FNC 160 /S1+ /S2+ /S3+ /S+ /D+	16	134,2	6,4	—
TZCP	FNC 161 /S1+ /S2+ /S+ /D+	16	140,2	6,4	—
TADD	FNC 162 /S1+ /S2+ /D+	16	118,8	6,4	—
TSUB	FNC 163 /S1+ /S2+ /D+	16	109,4	6,4	—
TRD	FNC 166 /D+	16	46,2	6,4	—
TWR	FNC 167 /S+	16	112	6,4	—
GRY	FNC 170 /S+	16	102,5	6,4	—
		32	107,1	6,4	
GBIN	FNC 171 /D+	16	103,4	6,4	—
		32	107,5	6,4	
LD□	FNC 221 – 230 /S1+ /S2+	16	1,52		—
		32	1,84		
AND□	FNC 232 – 238 /S1+ /S2+	16	1,52		—
		32	1,84		
OR□	FNC 240 – 246 /S1+ /S2+	16	1,52		—
		32	1,84		

Tab. B-25: Spezielle FNC-Anweisungen FX2N (2)

- ① „n“ gibt die Anzahl der statischen STL-Anweisungen (Anzahl der parallelen/zusammenführenden Anweisungen) an.
- ② Die gekennzeichneten Anweisungen benötigen keine Kontakte.
- ③ Wird die statische Anweisung eingesetzt und nicht die Pulsanweisung, verändert sich der Wert der Zieladresse.
- ④ „n“ gibt die Anzahl der zu verschiebenden Register an ($n \leq 512$).
- ⑤ „n“ gibt die Anzahl der zu verarbeitenden Bit-Operanden an ($n \leq 16$ im 16-Bit-Modus, $n \leq 32$ im 32-Bit-Modus).
- ⑥ „n“ gibt die Anzahl der zu verarbeitenden Bit-Operanden an.
- ⑦ „n“ gibt die Anzahl der zu verarbeitenden Operanden an ($2 \leq n \leq 512$).
- ⑧ „n“ gibt den zurückzusetzenden Operandenbereich an.
Der Operandentyp ist in den Klammern angegeben.
- ⑨ „n“ gibt die Operandenadressen an, die mit der MEAN-Anweisung verarbeitet werden ($1 \leq n \leq 64$).
- ⑩ „n“ gibt den zu aktualisierenden Operandenbereich an ($8 \leq n \leq 128$ in 8-ter-Schritten).
- ⑪ „n“ gibt die Zeitkonstante des Eingangsfilters an ($0 \leq n \leq 60$ ms).
- ⑫ Diese Anweisung kann max. 6 mal zur gleichen Zeit aktiv sein.
- ⑬ „n“ gibt die Anzahl der Stapel-Elemente an ($n \leq 256$ für 16-Bit-Verarbeitung, $n \leq 128$ für 32-Bit-Verarbeitung).
- ⑭ „n“ gibt die Anzahl der Ausgangsadressen an ($n \leq 64$).
- ⑮ „n“ gibt die Anzahl der Elemente der Datentabelle an ($1 \leq m1 \leq 32$). Zur vollständigen SORT-Verarbeitung wird die SORT-Anweisung $m1$ mal ausgeführt.
- ⑯ „n“ gibt die Anzahl der in das/aus dem Sondermodul zu schreibenden oder lesenden Datenworte an.
- ⑰ „n“ gibt die Anzahl der zu schreibenden oder lesenden Single-Byte-Datenworte (8 Bit) bei Parallelbetrieb zweier FX-Steuerungen an.

Index

!

32-Bit-Counter	3-18
High-Speed	3-16
16-Bit-Counter	
programmieren	3-12
rücksetzen	4-28
2-Phasen-Counter	3-20
32-Bit-Anweisungen	6-8
7-Segment-Anzeige	7-12

A

ABSD	6-121
ADD	6-46
Adressierung	
16-Bit-Counter	3-12
32-Bit-Counter	3-14
Ein-/Ausgänge	3-2
Interrupt-Pointer	3-40
Merker	3-5
Nesting	3-45
Pointer	3-39
Register	3-27
Schrittstatus	3-23
Allgemeine Systemdaten MELSEC FX1N . . .	A-7
Allgemeine Systemdaten MELSEC FX1S . . .	A-4
Allgemeine Systemdaten MELSEC FX2N . .	A-12
ALT	6-127
ANB	4-19
AND	4-9
AND (Vergleich)	7-101
ANF	4-15
ANI	4-9
ANP	4-15
ANR	6-86
ANS	6-85

Anweisungen	
32-Bit	6-8
Anwendungsbezogene	6-112
Anweisungen mit Gleitkommazahlen	7-50
Anweisungsliste	2-7
Anwendungsbereich	1-1
Übersicht	1-1
Anzeige-Bits	
rücksetzen	6-86
Anzeigemodul FX1N-5DM	8-12
Applikationsanweisungen	
Ausführung durch statisches Signal . .	6-6
Übersicht	6-9
Applikationsanweisungen FX1S/FX1N	A-10
Applikationsanweisungen FX2N	A-15
Arithmetische Anweisungen	6-45
ARWS	7-16
ASC	7-19
ASCII	7-35
Aufwärtszähler	4-44
Ausführungsregister für HSZ- und PLSY-Anweisungen	9-20
Ausführungszeiten	
Übersicht FX1S/FX1N	B-1
Übersicht FX2N	B-11
Ausgänge	3-2
adressieren	3-2
auffrischen	6-90
Beschreibung	3-2
Doppelbelegung	4-8
programmieren	3-4
Ausschalten beliebiger Interrupts	3-44
Auto-Tuning-Funktion	7-44

B

BCD-Anweisung	6-41
BCD-Format	3-38
Befehl	
Bedeutung	2-5
Beschaltung	
SPS	2-9
Beschaltung der SPS	2-9
BIN-Anweisung	6-43
Bit	6-3
Bit-Operanden	6-3
Bits	
ermitteln	6-82
Überprüfung	6-83
Bitweise verschieben	6-68
Block-Transfer	6-37
BMOV	6-37
BON	6-83
Borrow Flag	6-49
Byte	6-3

C

CALL-Anweisung	6-16
Carry Flag	6-49
CCD	7-39
CJ	6-12
CML	6-36
CMP	6-28
Counter	3-11
1-Phasen	3-18
2-Phasen	3-20
32-Bit	3-16
abs. Vergleich	6-121
Auf-/Abwärtszähler	9-11
High-Speed	3-16
rücksetzen	4-28

D

D	6-5
DABS	7-69
DADD	6-46
Daenübertragung	
seriell	7-28
DAND	6-56
DAND□	7-101
Daten	
codieren	6-80
decodieren	6-77
Datenausgabe	7-21
Datenaustausch	6-39
Datenaustausch mit FX0N-3A	7-95
Datenoperationen	6-75
Datenregister	3-25
Datenrotation	
Bits nach links	6-67
Bits nach rechts	6-66
nach links	6-65
nach rechts	6-64
Datenstruktur	6-5
Datentransfer	6-32
von gleichen Daten	6-38
Datenverarbeitungsanweisungen II	7-66
Datenverschiebung	
bitweise	6-68
wortweise nach links	6-71
wortweise nach rechts	6-70
Datenwort	6-3
DBCD	6-41
DBIN	6-43
DCMP	6-28
DCOS	7-64
DDEC	6-55
DDIV	6-52
DEADD	7-57
DEBCD	7-55

DEC	6-55	DZCP	6-30
DECMP	7-51	E	
DECO	6-77	Echtzeituhr-Anweisungen	7-78
DEDIV	7-60	Echtzeituhr-Funktion	8-9
Dekrementieren	6-55	EI	6-18
DEMUL	7-59	Eingang	
DESQR	7-61	Abfragebeispiel	4-36
Destination	6-5	Eingänge	3-2
DESUB	7-58	adressieren	3-2
DEZCP	7-53	auffrischen	6-90
Dezimalkonstante	3-24	Beschreibung	3-2
Dezimalzahlensystem	3-32	programmieren	3-4
DGBIN	7-94	EingangsfILTER	6-92
DGRY	7-93	einstellen	8-7
DHSCR	6-96	Eingangssignale	
DHSCS	6-96	Verarbeitung	3-3
DHSZ	6-98	Eingangssignalimpulse	
DI	6-18	kurzzeitige	8-5
Digitaler Schalter	7-10	ENCO	6-80
DINC	6-54	END	4-34
DINT	7-62	F	
DIV	6-52	Fehlercodes	10-3
DLD□	7-99	Fehlererkennung	10-1
DMOV	6-32	FEND-Anweisung	6-22
DMUL	6-50	FIFO-Speicher	
Doppelregister	3-25	aus Speicher lesen	6-73
DOR	6-58	in Speicher schreiben	6-72
DOR□	7-103	File-Register	8-10
DPLSR	6-110	Verwendung	3-30
DPLYS	6-106	Flag	
DRVA	7-76	Bedeutung	6-8
DRVI	7-74	Sonderregister	9-16
DSIN	7-63	Übersicht	9-4
DSUB	6-48	Fließkomma-Zahlensystem	3-34
DSW	7-10	Flip-Flop-Funktion	6-127
DSWAP	7-67	FLT	6-88
DTAN	7-65	Flußdiagramm	5-4
Dualzahlensystem	3-33		
DXOR	6-60		

FMOV 6-38
 FNC-Anweisungen 7-1
 FOR 6-25
 FROM 7-23
 Funktionsplan 2-7

G

GBIN 7-94
 Geschwindigkeitserkennung 6-104
 Gleitkommazahlen 7-50
 Gray-Code-Anweisungen 7-92
 Grundbefehle
 Übersicht A-1
 Grundbefehlssatz 4-1
 Ausführungszeiten B-1
 Übersicht 4-2
 GRY 7-93

H

Hauptkontakt
 programmieren 4-24
 HEX 7-37
 Hexadezimalkonstante 3-24
 Hexadezimalsystem 3-37
 High-Speed-Anweisungen 6-89
 High-Speed-Counter 3-16
 setzen und rücksetzen 6-96
 Zähleingänge 3-16
 Zählrichtung wählen 9-11
 Hilfsrelais 3-5
 HKY 7-7

I

Impulserzeugung 4-29
 Ausgabe 6-106
 Ausgabe (PWM) 6-108
 Impulssignal 6-6
 Impulsweitenmodulation 6-108
 INC 6-54

INCD 6-123
 Indexregister 3-29
 Einsatz 6-7
 Index-Register 9-21
 Inkrementaler Counter-Vergleich 6-123
 Inkrementieren 6-54
 INT 7-62
 Interrupt-Pointer 3-40
 Interrupt-Programm
 Einsatz 6-18
 INV 4-31
 Invertierfunktion 6-36
 IRET 6-18
 IST 6-113

K

Kommunikationsadapter 9-19
 Kommunikationsparameter 7-29
 Kontaktplan 2-8
 Kontrollbedingung
 setzen und rücksetzen 4-24
 Konvertierung
 ASCII 7-19
 BCD 6-41
 Binär 6-43
 Kopierfunktion 6-36

L

Lade 4-5
 LD 4-5
 LD/LDI-Anweisung 4-5
 LDF 4-13
 LDI 4-5
 LD□ 7-99
 LDP 4-13
 Link- und Sonderfunktionen 9-18
 Link-Funktionen 9-9
 Logische Verknüpfungen 6-56

M

Master Control	4-24
Master Control Reset	4-24
MC	6-14
MC-, MCR-Anweisung	4-24
MCR	4-24
Sprünge im MC-Bereich	6-14
MEAN	6-84
Merker	3-5
adressieren	3-5
Beschreibung	3-5
programmieren	3-6
MOV	6-32
MPP	4-21
MPS	4-21
MPS-, MRD-, MPP-Anweisung	4-21
MRD	4-21
MTR	6-93
MUL	6-50

N

NEG	6-62
Nesting	3-45
programmieren	4-24
Nesting-Ebenen	3-39
NEXT	6-25
NOP	4-32
numerische Daten	
addieren	6-46
Numerische Daten	
dividieren	6-52
multiplizieren	6-50
subtrahieren	6-48
vergleichen	6-28
Numerische Datenbereiche	
vergleichen	6-30

O

ODER-Verknüpfung	6-58
Oktalzahlsystem	3-36
Operanden	3-1
Ausgänge	3-2
Beschreibung	6-3
Counter	3-11
Dezimalkonstante	3-24
Einführung	2-6
Eingänge	3-2
Erläuterung	2-6
Hexadezimalkonstante	3-24
Interrupt-Pointer	3-40
Merker	3-5
Nesting	3-45
Pointer	3-39
Register	3-25
Schrittstatus	3-23
Setzen und Rücksetzen	4-27
Sondermerker	9-1
Sonderregister	9-14
Timer	3-7
Übersicht	3-1
Übersicht FX1N	A-8
Übersicht FX1S	A-5
Übersicht FX2N	A-13
Operandenbereiche	
zurücksetzen	6-76
OR	4-11
OR-/ORI-Anweisung	4-11
ORB	4-20
ORF	4-17
ORI	4-11
OR□	7-103
ORP	4-17
OUT	4-7

P

Paritätsprüfung	7-39
Passwortfunktion	8-4
PID	7-43
PLF	4-29
PLS	4-29
PLSR	6-110
PLSV	7-72
PLSY	6-106
Pointer	3-39
programmieren	6-12
Pointeradresse	6-13
Pointermarkierung	6-12
Positionieranweisungen bei FX1S/FX1N	7-68
Potentiometer bei FX1S/FX1N	
Beschreibung	8-8
Vorgabe eines Zeitsollwertes	3-9
PR	7-21
Programm	
Interrupt	9-7
Leerzeile im	4-32
Sprung in einem	6-12
Unterprogramm-Ende	6-17
Programmabarbeitung	2-1
Programmablaufanweisungen	6-11
Programmbereichsende	6-22
Programmende	4-34
Programmfehler	10-1
Programmierung	
Grundlagen	2-1
Programmschleife	6-25
Programmschritte	6-2
Programmverarbeitung	2-1
Programmwiederholung	6-25
Programmzykluszeit	
konstante	8-3
maximale	6-23
Prozessabbildverfahren	2-2

PRUN	7-34
Pulse-Catch-Funktion	8-5
Sondermerker	9-8
PWM	6-108

Q

Quadratwurzel	6-87
Quelldaten	6-5

R

RAMP	6-129
Rampenfunktion	6-129
RCL	6-67
RCR	6-66
RD3A	7-96
REF	6-90
REFF	6-92
Regelkreis, geschlossen	7-43
Register	3-25
adressieren	3-27
Aufbau	3-26
Registertypen	3-25
Remanente Timer	3-10
RET	5-5
ROL	6-65
ROR	6-64
ROTC	6-131
RS	7-28
RST	4-27
Rücksetzen von Operanden	4-27
RUN-/STOP-Umschaltung	8-11

S

Schrittstatus	3-23
adressieren	3-23
initialisieren	5-10
initialisieren (IST)	6-113
Schrittstatusoperanden	5-1
Beschreibung	3-23

Schrittsteueranweisungen	5-1, 5-5	STL-Status	9-17
programmieren	5-5	Übersicht	9-14
Schrittsteuerung		Verwendung	3-27
Ausgänge mehrfach belegen	5-7	Zeittakt	9-16
Flußdiagramm	5-4	Sonstige Sonderregister	9-18
Schematischer Ablauf	5-3	SORT	6-134
Timer mehrfach belegen	5-8	Sortieranweisung	6-134
Schutzebenen	8-4	Source	6-5
SEGD	7-12	SPD	6-104
SEGL	7-13	Sprung	6-12
SER	6-119	SPS-Modus	
Serielle Kommunikation	7-27	Sondermerker	9-5
SET	4-27	Sonderregister	9-17
Setzen von Operanden	4-27	SPS-Status	
SFRD	6-73	Sondermerker	9-2
SFTL	6-68	Sonderregister	9-15
SFTR	6-68	SQR	6-87
SFWR	6-72	SRET-Anweisung	6-17
Shift-Transfer	6-33	Statisches Signal	6-6
SMOV	6-33	Steuerungsanweisungen	2-5
Sonderfunktionen	8-1	Aufbau	2-5
Sondermerker	9-1	Darstellungsarten	2-7
Fehlererkennung	10-1	STL-Anweisung	5-1
Flag	9-4	STL-Status	
High-Speed-Counter	9-11	Sondermerker	9-6
Interrupt-Programm	9-7	Sonderregister	9-17
Pulse-Catch-Funktion	9-8	STI-Verzweigungen	
SPS-Modus	9-5	selektive	5-12
SPS-Status	9-2	STL-Verzweigungen	5-11
STL-Status	9-6	Einfachverlauf	5-11
Sondermodul		Leerstatus	5-17
Daten auslesen	7-23	parallele	5-14
Daten schreiben	7-25	Sprung	5-18
Sonderregister		STMR	6-126
Fehlererkennung	10-2	STOP-Modus	
Flag	9-16	Datenerhalt im	8-2
für serielle Kommunikation	9-19	SUB	6-48
SPS-Modus	9-17	Suchanweisung	6-119
SPS-Status	9-15	SUM	6-82

Summenprüfung	7-39
SWAP	7-67
Systemdaten	
FX1S	A-4
FX1N	A-7
FX2N	A-12

T

TADD	7-83
Tastatur	
hexadezimal	7-7
zehner	7-5
TCMP	7-79
Technische Daten	
Systemdaten FX1N	A-7
Systemdaten FX1S	A-4
Systemdaten FX2N	A-12
Timer	3-7
Beschreibung	3-7
Genauigkeit	3-9
Programmierbeispiel	4-42
Spezial	6-126
Teaching	6-125
Watch Dog	6-23
Wertvorgabe mit Potentiometer	3-9
TKY	7-5
TO	7-25
Transferanweisungen	6-27
TRD	7-87
TSUB	7-85
TTMR	6-125
TWR	7-89
TZCP	7-81

U

Umwandlung	
ASCII-Code	7-35
hexadezimal	7-37
Umwandlung des Zahlenformats	6-88

Unterprogramm

Aufruf	6-16
Ende	6-17

V

Vergleich

numerische Daten	6-28
------------------	------

Vergleichsanweisungen II	7-98
--------------------------	------

Verknüpfungen

ODER	4-11
ODER-Block	4-20
UND	4-9
UND-Block	4-19

Verknüpfungsbeginn	4-5
--------------------	-----

Verknüpfungsebene	4-21
-------------------	------

Verknüpfungsergebnis

abspeichern	4-21
ausgeben	4-7

Verschiebeanweisungen	6-63
-----------------------	------

VRRD	7-41
------	------

VRSC	7-42
------	------

W

WAND	6-56
------	------

WDT-Anweisung	6-23
---------------	------

Weiterschaltbedingung	5-7
-----------------------	-----

Wiederholung

Programmteile	6-25
---------------	------

WOR	6-58
-----	------

Wort	6-3
------	-----

Wortoperanden	6-3
---------------	-----

Wortverarbeitung	6-3
------------------	-----

WR3A	7-97
------	------

WSFL	6-71
------	------

WSFR	6-70
------	------

WXOR	6-60
------	------

X

XCH 6-39

Z

Zahlen im wissenschaftlichen Format 3-33

Zahlendarstellungen 3-32

Zahlenwertebereich

dezimal 3-24

hexadezimal 3-24

Zähler 3-11

Zählersollwert 3-12

Zählersollwertangabe

direkt 3-13

indirekt 3-13

ZCP 6-30

Zehnertastatur 7-5

Zeitglieder 3-7

Zeitintervall

starten 6-85

Zeitsollwert 3-8

Zeitsollwertangabe

direkt 3-8

indirekt 3-8

Zeittakt

Sondermerker 9-3

Sonderregister 9-16

Zero Flag 6-49

Zieldaten 6-5

ZRN 7-70

ZRST 6-76

Zuordnungsliste 2-9

HEADQUARTERS

MITSUBISHI ELECTRIC EUROPE B.V. **EUROPA**
 German Branch
 Gothaer Straße 8
D-40880 Ratingen
 Telefon: +49 (0) 21 02 / 486-0
 Telefax: +49 (0) 21 02 / 4 86-1 12
 E-Mail: megfa-mail@meg.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. **ITALIEN**
 Italian Branch
 C.D. Colleoni - P. Perseo Ing. 2
 Via Paracelso 12
I-20041 Agrate Brianza (MI)
 Telefon: +39 039 6053 1
 Telefax: +39 039 6053 312
 E-Mail: industrial@it.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. **SPANIEN**
 Pol. Ind. Can Magi
 Calle Joan Buscallá, 2-4 AC 420
E-08190 Sant Cugat del Vallés
 Telefon: +34 (9) 3 / 565 31 60
 Telefax: +34 (9) 3 / 589 15 79
 E-Mail: industrial@sp.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. **UK**
 UK Branch
 Travellers Lane
GB-Hatfield Herts. AL10 8 XB
 Telefon: +44 (0) 1707 / 27 61 00
 Telefax: +44 (0) 1707 / 27 86 95

MITSUBISHI ELECTRIC CORPORATION **JAPAN**
 Mitsubishi Denki Bldg.
 2-2-3 Marunouchi Chiyoda-Ku
Tokyo 100-8310
 Telefon: +81 (0) 3 / 32 18 31 76
 Telefax: +81 (0) 3 / 32 18 24 22

MITSUBISHI ELECTRIC AUTOMATION **USA**
 500 Corporate Woods Parkway
Vernon Hills, IL 60061
 Telefon: +1 847 / 478 21 00
 Telefax: +1 847 / 478 22 83

EUROPÄISCHE VERTRETUNGEN

Getronics Ind. Autom. bv. **BELGIEN**
 Control Systems
 Pontbeeklaan 43
B-1731 Asse-Zellik
 Telefon: +32 (0) 2 / 4 67 17 51
 Telefax: +32 (0) 2 / 4 67 17 45
 E-Mail: j.vanhaute@getronics.be

TELECON CO. **BULGARIEN**
 4, A. Ljapchev Blvd.
BG-1756 Sofia
 Telefon: +359 92 / 97 44 05 8
 Telefax: +359 92 / 97 44 06 1
 E-Mail: —

louis poulsen **DÄNEMARK**
 Geminivej 32
DK-2670 Greve
 Telefon: +45 (0) 43 / 95 95 95
 Telefax: +45 (0) 43 / 95 95 91
 E-Mail: lpia@lpmail.com

UTU Elektrotehnika AS **ESTLAND**
 Laki 12E
EE-10621 Tallinn
 Telefon: +372 6 / 517 280
 Telefax: +372 6 / 517 288
 E-Mail: utu@uninet.ee

Beijer Electronics OY **FINNLAND**
 Elannontie 5
FIN-01510 Vantaa
 Telefon: +358 (0) 9 / 615 20 11
 Telefax: +358 (0) 9 / 615 20 500
 E-Mail: info@elc.beijer.fi

IP Systèmes **FRANKREICH**
 8, Rue du Colonel Chambonnet
F-69672 Lyon Bron Cedex
 Telefon: +33 (0) 4 / 72 14 18 00
 Telefax: +33 (0) 4 / 72 14 18 01
 E-Mail: info@ip-systemes.fr

MITSUBISHI ELECTRIC EUROPE B.V. – Irish Branch **IRLAND**
 Westgate Business Park
IRL-Dublin 24
 Telefon: +353 (0) 1 / 419 88 00
 Telefax: +353 (0) 1 / 419 88 90
 E-mail: sales.info@meuk.mee.com

Getronics Ind. Autom. bv **NIEDERLANDE**
 Control Systems
 Donauweg 10
NL-1043 AJ-Amsterdam
 Telefon: +31 (0) 20 / 586 15 92
 Telefax: +31 (0) 20 / 586 19 27
 E-Mail: info.algemeen@getronics.nl

Beijer Electronics A/S **NORWEGEN**
 Teglværksveien 1
N-3002 Drammen
 Telefon: +47 (0) 32 / 24 30 00
 Telefax: +47 (0) 32 / 84 85 77
 E-Mail: info@elc.beijer.no

EUROPÄISCHE VERTRETUNGEN

GEVA GmbH **ÖSTERREICH**
 Wiener Straße 89
A-2500 Baden
 Telefon: +43 (0) 2252 / 85 55 20
 Telefax: +43 (0) 2252 / 488 60
 E-Mail: office@geva.co.at

MPL Technology SP. z.o.o **POLEN**
 ul. Wroclawska 53
PL-30011 Kraków
 Telefon: +48 (0) 12 / 632 28 85
 Telefax: +48 (0) 12 / 632 47 82
 E-Mail: mpl@krakow.ipl.net

Sirius Trading & Services srl **RUMÄNIEN**
 Bd. Ghica nr. 112, Bl. 41
RO-72335 Bucaresti 2
 Telefon: +40 (0) 1 / 210 55 11
 Telefax: +40 (0) 1 / 210 55 11
 E-Mail: sirius_t_s@fx.ro

Beijer Electronics AB **SCHWEDEN**
 Box 325
S-20123 Malmö
 Telefon: +46 (0) 40 / 35 86 00
 Telefax: +46 (0) 40 / 93 23 01
 E-Mail: info@elc.beijer.se

ECONOTEC AG **SCHWEIZ**
 Postfach 282
CH-8309 Nürensdorf
 Telefon: +41 (0) 1 / 838 48 11
 Telefax: +41 (0) 1 / 838 48 12
 E-Mail: info@econotec.ch

ACP AUTOCOMP a.s. **SLOWAKEI**
 Chalupkova 7
SK-81109 Bratislava
 Telefon: +421 (0) 7 592 22 48
 Telefax: +421 (0) 7 592 22 54
 E-Mail: acp.autocomp@nexta.sk

INEA d.o.o. **SLOWENIEN**
 Ljubljanska 80
SI-61230 Domžale
 Telefon: +386 (0) 1 / 721 80 00
 Telefax: +386 (0) 1 / 724 16 72
 E-Mail: zoran.marinsek@inea.si

AUTOCONT s.r.o. **TSCHECHIEN**
 Nemocnici 12
CZ-70100 Ostrava 1
 Telefon: +420 (0) 69 / 615 21 11
 Telefax: +420 (0) 69 / 615 21 12
 E-Mail: petr.pustovka@autocont.cz

GTS **TÜRKEI**
 Darülaceze Cad. No. 43A KAT: 2
TR-80270 Okmeydani-Istanbul
 Telefon: +90 (0) 212 / 320 1640
 Telefax: +90 (0) 212 / 320 1649
 E-Mail: gts@turk.net

VERKAUFSBÜROS DEUTSCHLAND

MITSUBISHI ELECTRIC EUROPE B.V. **DEUTSCHLAND**
 DGZ-Ring Nr. 7
D-13086 Berlin
 Telefon: (0 30) 4 71 05 32
 Telefax: (0 30) 4 71 54 71

MITSUBISHI ELECTRIC EUROPE B.V. **DEUTSCHLAND**
 Revierstraße 5
D-44379 Dortmund
 Telefon: (02 31) 96 70 41-0
 Telefax: (02 31) 96 70 41-41

MITSUBISHI ELECTRIC EUROPE B.V. **DEUTSCHLAND**
 Brunnenweg 7
D-64331 Weiterstadt
 Telefon: (0 61 50) 13 99 0
 Telefax: (0 61 50) 13 99 99

MITSUBISHI ELECTRIC EUROPE B.V. **DEUTSCHLAND**
 Kurze Strasse 40
D-70794 Filderstadt-Bonlanden
 Telefon: (07 11) 77 05 98-0
 Telefax: (07 11) 77 05 98-79

MITSUBISHI ELECTRIC EUROPE B.V. **DEUTSCHLAND**
 Am Söldnermoos 8
D-85399 Hallbergmoos
 Telefon: (08 11) 99 87 4-0
 Telefax: (08 11) 99 87 4-10

VERTRETUNG MITTLERER OSTEN

TEXEL Electronics Ltd. **ISRAEL**
 PO Box 6272
IL-Netanya 42160
 Telefon: +972 (0) 9 / 863 08 94
 Telefax: +972 (0) 9 / 885 24 30
 E-Mail: texel_me@netvision.net.il

ILAN & GAVISH LTD **ISRAEL**
 24 Shenkar St., Qiryat-Arie 49513
IL-49001 Petach-Tikva
 Telefon: +972 (0) 3 / 922 18 24
 Telefax: +972 (0) 3 / 972 39 24 07 61
 E-mail: iandg@internet-zahav.net

VERTRETUNGEN EURASIEN

MITSUBISHI ELECTRIC EUROPE B.V. **RUSSLAND**
 12/1 Goncharnaya St, suite 3C
RUS-109240 Moskow
 Telefon: +7 (0) 95 / 915-8602
 Telefax: +7 (0) 95 / 915-8603

NPP Uralelektra **RUSSLAND**
 Sverdlova 11A
RUS-620027 Ekaterinburg
 Telefon: +7 34 32 / 53 27 45
 Telefax: +7 34 32 / 53 24 61
 E-mail: elektra@etel.ru

JV-CSC Automation **UKRAINE**
 15, Marina Raskovoyi St.
U-02002 Kiev
 Telefon: +380 (44) / 238 83 16
 Telefax: +380 (44) / 238 83 17
 E-Mail: mkl@csc-a.kiev.ua