SIEMENS

SIMATIC

S7-300 および S7-400 のプログラミング用ラダーロジック (LAD)

リファレンスマニュアル

05/2017

A5E41733445-AA

法律上の注意

警告事項

本書には、ユーザーの安全を守るため、および製品や接続された機器の損傷を防ぐために遵守すべき注意事項が記載されています。ユーザーの安全性に関する注意事項は、安全警告サインで強調表示されています。このサインは、物的損傷に関する注意事項には表示されません。

⚠危険

適切な予防措置を講じなければ、きわめて高い可能性で、死亡、重傷、または機器の重大な損傷を引き起こす恐れがあります。

⚠警告

適切な予防措置を講じない場合、死亡、重傷、または機器の重大な損傷を引き起こす恐れがあります。

⚠注意

適切な予防措置を講じなければ、人体に軽度の傷害を引き起こす恐れがあります。

注意

適切な予防措置を講じなければ、機器の損傷を引き起こす恐れがあります。

複数の危険度が存在する場合、一番高い危険度を表す警告が使用されます。安全警告シンボルを伴う人的傷害に関する警告には、物的損害に関する警告も含まれる場合があります。

有資格者

装置/システムのセットアップおよび使用にあたっては必ず本マニュアルを参照してください。機器のインストールおよび操作は有資格者のみが行うものとします。有資格者とは、法的な安全規制/規格に準拠してアースの取り付け、電気回路、設備およびシステムの設定に携わることを承認されている技術者のことをいいます。

シーメンス製品を正しくお使いいただくために

以下の点に注意してください。

⚠警告

シーメンス製品は、カタログおよび付属の技術説明書の指示に従ってお使いください。他社の製品または部品との併用は、弊社の推奨もしくは許可がある場合に限ります。シーメンス 製品を正しく安全にご使用いただくには、適切な運搬、保管、組み立て、据え付け、配線、始動、操作、保守を行ってください。ご使用になる場所は、許容された範囲を必ず守ってください。付属の技術説明書に記述されている指示を遵守してください。

商標

本書において®で識別されるすべての名称は、Siemens AGの登録商標です。その他、この文書に記載されている会社名や製品名は各社の商標であるため、第三者が自己の目的のためにこれらの名前を使用すると、商標所有者の権利を侵害する恐れがあります。

免責事項

本書の内容は、記載されているハードウェアやソフトウェアとの齟齬がないよう見直されています。しかしながら、相違点をすべて取り除くことはできないため、完全な一致を保証するものではありません。本マニュアルの内容は定期的に見直され、必要な修正は次回の版で行われます。

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
DEUTSCHLAND

A5E41733445-AA
Ⓟ 05/2017予告なしに変更することがあります

はじめに

目的

このマニュアルは、ステートメントリストプログラミング言語ラダーロジックのユーザプログラムを作成するための手引きです。

ラダーロジックの言語要素の構文、ファンクションを説明する参照ページも含まれています。

必要な基本知識

このマニュアルの対象者は、S7 プログラマ、オペレータ、保守/サービス要員です。

本マニュアルを理解するには、自動化技術の全般的な知識が必要です。

さらに、コンピュータについての知識、オペレーティングシステム MS Windows XP、MS Windows Server 2003、MS Windows 7 の下での PC(プログラミングデバイス等)同様やその他の動作機器の知識が必須です。

本マニュアルの対応バージョン

本マニュアルは STEP 7 プログラミングソフトウェアパッケージの 5.6 版用です。

標準との適合

LAD は IEC(国際電気標準会議)-1131-3 で定義された「ラダーロジック」に対応します。ただし、運用に関してはかなりの差異があります。詳細は、STEP 7 ファイル NORM_TBL.RTF の標準表を参照してください。

オンラインヘルプ

本マニュアルはソフトウェアに組み込まれているオンラインヘルプと併せてご使用ください。このオンラインヘルプは、STEP 7 を使用する際に詳細なサポートを提供することを目的としています。

ソフトウェアに組み込まれたヘルプシステムでは、いくつかのインターフェースが利用できます。

- 状況に応じたヘルプは、現在のコンテキスト、たとえば開いたダイアログ、アクティブなウィンドウについての情報を提供します。状況に応じたヘルプは、メニューコマンドの[ヘルプ|状況に応じたヘルプ]を選択するか、[F1]キーを押すか、ツールバーの疑問符マークの記号を使います。
- [STEP 7 のヘルプ]を呼び出すには、メニューコマンドから[ヘルプ|目次]または状況に応じたヘルプウィンドウの[STEP 7 のヘルプ]ボタンを使用します。
- [用語集]ボタンで、STEP 7 アプリケーションの用語集を呼び出すことができます。

このマニュアルは、「ステートメントリストのヘルプ」からの抜粋です。マニュアルとオンラインヘルプがほぼ同一構造なので、マニュアルとオンラインヘルプを交互に容易に利用できます。

その他のサポート

技術的な質問がある場合、シーメンスの担当者または代理店の担当者に連絡をとってください。

連絡先は下記のアドレスで検索できます。

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

各 SIMATIC 製品およびシステムの技術文書のガイドは、以下でご覧になれます。

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

オンラインカタログおよび注文システムは以下にあります。

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

トレーニングセンター

Siemens 社は、SIMATIC S7 オートメーションシステムに精通していただくためのたくさんのトレーニングコースを用意しております。詳しくは、該当地区のトレーニングセンターか、下記のドイツ D 90026 ニュルンベルクの中央トレーニングセンターにご連絡ください。

Internet: http://sitrain.automation.siemens.com/sitrainworld/

技術サポート

すべての産業用オートメーション&ドライブテクノロジ製品のテクニカルサポートは次のとおりです。

- サポートリクエスト Web フォーム経由

http://www.siemens.com/automation/support-request

テクニカルサポートについての追加情報は、インターネットの

http://www.siemens.com/automation/serviceページにあります。

インターネットによるサービスとサポート

マニュアルの他に、ノウハウを次のインターネットで提供します。

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

内容:

- ニュースレター、製品に関する最新情報を提供
- 「Service & Support」の検索機能を利用して必要な文書を検索
- フォーラム、世界中のユーザや専門家がその経験を交換
- 産業用オートメーション&ドライブテクノロジを担当するお客様の最寄りのお問い合わせ先
- フィールドサービス、修理、スペアパーツ、コンサルティングについての情報

セキュリティ情報：

シーメンスの製品およびソリューションでは、プラント、システム、マシン、ネットワークの安全な操作をサポートする産業用セキュリティファンクションが提供されています。

プラント、システム、マシン、ネットワークをサイバー攻撃から保護するには、総合的な最新の産業用セキュリティコンセプトを実装し、継続的に維持する必要があります。シーメンスの製品およびソリューションのみが、このようなコンセプトの 1 つのエレメントを形成します。

お客様のプラント、システム、マシン、ネットワークへの未許可のアクセスを防止するのは、お客様の責任です。システム、マシン、コンポーネントは、必要な場合に必要な部分のみ、インターネットの企業ネットワークに接続します。この場合、適当な位置で適切なセキュリティ手段(たとえば、ファイアウォールおよびネットワークセグメンテーションの使用など)を使用します。

さらに、適切なセキュリティ手段に関するシーメンスのアドバイスを検討する必要があります。産業用セキュリティに関する詳細は、次を参照してください

http://www.siemens.com/industrialsecurity。

シーメンスの製品およびソリューションは、セキュリティ度を高めるための継続的な更新を続けます。製品更新が使用可能になったらすぐにそれを適用し、常に最新の製品バージョンを使用することをシーメンスは強く推奨します。サポートされなくなった製品バージョンを使用したり、最新の更新を適用しないままでいると、お客様のサーバー攻撃に曝される度合いが高まります。

製品更新に関する継続通知を希望する場合は、次の Siemens Industrial Security RSS Feed でその申し込みを行います

http://www.siemens.com/industrialsecurity。

目次

1 ビットロジック命令

1.1 ビット論理命令の概要

説明

ビット論理命令は、1 と 0 の 2 つの桁で動作します。これらの 2 つの桁は、2 進法という記数法の基本となります。1 と 0 の 2 つの数字は、2 進数またはビットと呼ばれます。接点やコイルにおいて、1 は有効または通電状態であることを示し、0 は有効ではない、通電状態ではないことを示します。

ビット論理命令は 1 と 0 の信号状態を解読し、ブールロジックに従ってそれらを結合します。これらの結合により、"論理演算結果"(RLO)と呼ばれる 1 または 0 の結果が生成されます。

ビット論理命令によってトリガされる論理演算は、さまざまな機能を実行します。

次のファンクションを実行するビット論理命令があります。

- ---| |--- a 接点(アドレス)
- ---| / |--- b 接点(アドレス)
- ---(SAVE) RLO を BR メモリに保存
- XOR ビット排他的論理和
- ---( ) 出力コイル
- ---(#)--- 中間出力
- ---|NOT|--- パワーフローの反転

次の命令は、1 の RLO に対して反応します。

- ---(S) セットコイル
- ---(R) リセットコイル
- SR セット-リセットフリップフロップ
- RS リセット-セットフリップフロップ

その他の命令は信号立ち上がりまたは信号立ち下がりへの移行に反応して、次のファンクションを実行します。

- ---(N)--- 立ち下がり RLO 信号検出
- ---(P)--- 立ち上がり RLO 信号検出
- NEG アドレス立ち下がりパルス
- POS アドレス立ち上がり信号検出

- 即時読み取り
- 即時書き込み

1.2 ---| |--- a 接点(アドレス)

シンボル

<address>

---| |---

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D、T、C	チェック済みビット

説明

---| |--- (a 接点)は、指定した**<アドレス>**に格納されたビット値が"1"になると閉じます。接点が閉じると、ラダー母線の電流が接点を流れ、論理演算(RLO)の結果は 1 になります。

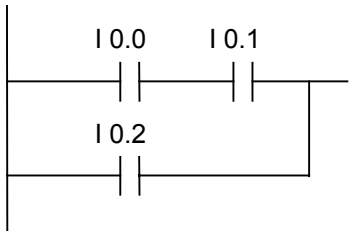
一方、指定された**<address>** の信号状態が"0"になると、接点は開きます。接点が開くと、接点に電流が流れず、論理演算の結果(RLO)は"0"になります。

直列接続で使用すると、---| |---は AND 論理により RLO ビットにリンクされます。並列接続で使用すると、OR 論理により RLO ビットにリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	X	X	X	1

例



次のどちらかの状態になると、電流が流れます。

入力 I0.0 と I0.1 の信号状態が"1"の場合

入力 I0.2 の信号状態が"1"の場合

1.3 ---| / |--- b 接点(アドレス)

シンボル

<address>

--| / |---

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D、T、C	チェック済みビット

説明

---| / |--- (b 接点)は、指定した**<アドレス>**に格納されたビット値が"0"になると閉じます。接点が閉じると、ラダー母線の電流が接点を流れ、論理演算(RLO)の結果は 1 になります。

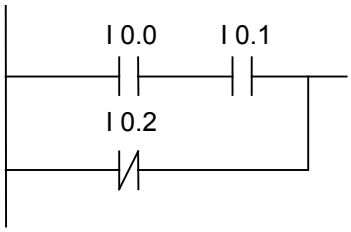
一方、指定された**<address>**の信号状態が"1"になると、接点は開きます。接点が開くと、接点に電流が流れず、論理演算の結果(RLO)は"0"になります。

直列接続で使用すると、---| / |---は AND 論理により RLO ビットにリンクされます。並列接続で使用すると、OR 論理により RLO ビットにリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	X	X	X	1

例



次のどちらかの状態になると、電流が流れます。

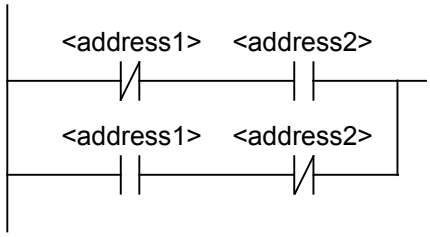
入力 I0.0 と I0.1 の信号状態が"1"の場合

入力 I0.2 の信号状態が"0"の場合

1.4 XOR ビット排他的論理和

XOR ファンクションでは、通常開いている接点と閉じている接点を次のように接続する必要があります。

シンボル

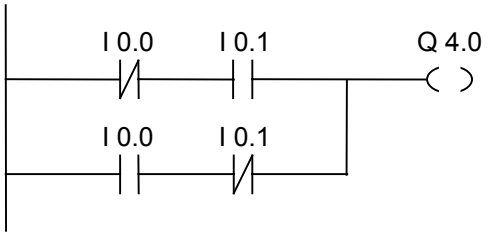


パラメータ	データタイプ	メモリ領域	説明
<address1>	BOOL	I、Q、M、L、D、T、C	スキャン済みビット
<address2>	BOOL	I、Q、M、L、D、T、C	スキャン済みビット

説明

XOR (ビット排他的論理和)では、2 つの指定されたビットの信号状態が異なる場合、RLO は"1"になります。

例



I0.0 が"0"で I0.1 が"1"の場合、または I0.0 が"1"で I0.1 が"0"の場合、出力 Q4.0 は"1"になります。

1.5 --|NOT|-- パワーフローの反転

シンボル

---|NOT|---

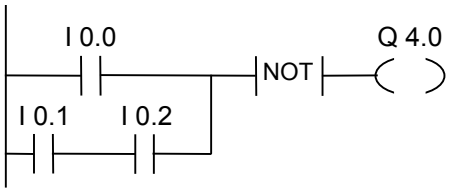
説明

---|NOT|--- (パワーフローの反転)は、RLO ビットを否定します。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	-	1	X	-

例



次のどちらかの状態になると、出力 Q4.0 の信号状態は"0"になります。

入力 I0.0 の信号状態が"1"の場合

入力 I0.1 と I0.2 の信号状態が"1"の場合

1.6 ---( ) 出力コイル

シンボル

<address>

---( )

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	割り付け済みビット

説明

---( ) (出力コイル)は、リレー論理図のコイルと同様に機能します。コイルへの電流の流れが存在する場合(RLO=1)、位置**<アドレス>**のビットは"1"に設定されます。コイルへの電流の流れが存在しない場合(RLO=0)、位置**<アドレス>**のビットは 0 に設定されます。出力コイルは、ラダー回路の右側にのみ配置できます。複数の出力エレメント(最大 16)が可能です(例を参照)。出力を否定する場合は、---|NOT|--- (パワーフローの反転)エレメントを使用します。

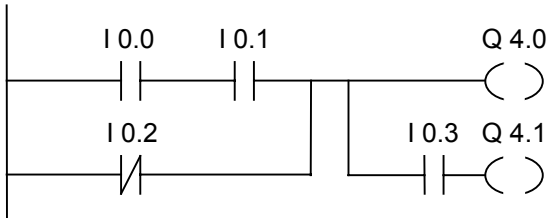
MCR (マスタコントロールリレー)の依存性

MCR の依存性は、アクティブな MCR ゾーンに出力コイルを配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになり、かつ出力コイルへ電流が流れると、アドレス指定されたビットが、パワーフローの現在の状態に設定されます。MCR がオフの場合は、パワーフローの状態に関係なく、指定されたアドレスに論理"0"が書き込まれます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	X	-	0

例



次のどちらかの状態になると、出力 Q4.0 の信号状態は"1"になります。
入力 I0.0 と I0.1 の信号状態が"1"の場合
入力 I0.2 の信号状態が"0"の場合
次のどちらかの状態になると、出力 Q4.1 の信号状態は"1"になります。
入力 I0.0 と I0.1 の信号状態が"1"の場合
入力 I0.2 の信号状態が"0"で、入力 I0.3 の信号状態が"1"の場合

例に示す回路がアクティブな MCR ゾーン内にある場合、以下が実行されます。
MCR がオンの場合、前述のように、パワーフローの状態に従って Q4.0 と Q4.1 が設定されます。
MCR がオフの場合(=0)、パワーフローの状態に関係なく、Q4.0 と Q4.1 は 0 にリセットされます。

1.7 ---(#)--- 中間出力

シンボル

<address>
---(#)---

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I, Q, M, *L, D	割り付け済みビット

*L 領域アドレスを使用するには、論理ブロック(FC、FB、OB)の変数宣言テーブルで TEMP と宣言する必要があります。

説明

---(#)--- (中間出力)は、指定された **<address>**に、RLO ビット(パワーフローの状態)を保存する中間割り付けエレメントです。中間出力エレメントにより、前に実行された分岐エレメントの論理結果が保存されます。他の接点と直列に、---(#)---は、接点と同様に挿入されます。---(#)---エレメントは、制御母線に接続したり、分岐接続の直後や分岐の終わりに接続することはできません。---(#)---を否定する場合は、---|NOT|---(パワーフローの反転)エレメントを使用します。

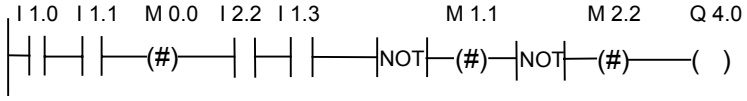
MCR (マスタコントロールリレー)の依存性

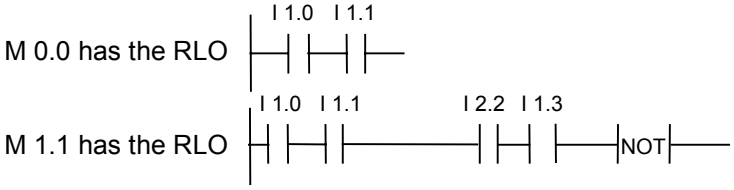
MCR の依存性は、アクティブな MCR ゾーンに中間出力コイルを配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになり、かつ中間出力コイルへ電流が流れると、アドレス指定されたビットが、パワーフローの現在の状態に設定されます。MCR がオフの場合は、パワーフローの状態に関係なく、指定されたアドレスに論理"0"が書き込まれます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	X	-	1

例





M 2.2 has the RLO of the entire bit logic combination

1.8 ---(R) リセットコイル

シンボル

<address>

---(R)

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D、T、C	リセットビット

説明

---(R) (リセットコイル)は、前に実行された命令の RLO が"1"の場合に実行されます(コイルへのパワーフロー)。コイルへの電流の流れが存在する場合(RLO="1")、指定したエレメントの**<アドレス>**は"0"にリセットされます。RLO が"0"(コイルへの電流の流れなし)の場合は効果はなく、指定したエレメントの状態は変わりません。**<address>**は、タイマ値が"0"リセットされるタイマ(T 番号)、またはカウンタ値が"0"リセットされるカウンタ(C 番号)にもなります。

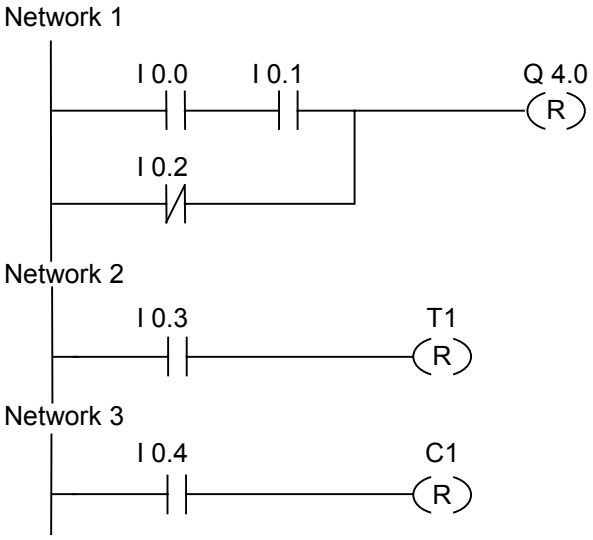
MCR (マスタコントロールリレー)の依存性

MCR の依存性は、アクティブな MCR ゾーンにリセットコイルを配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになり、かつリセットコイルへ電流が流れると、アドレス指定されたビットが"0"に設定されます。MCR がオフの場合、パワーフローの状態に関係なく、エレメントに対し指定されたアドレスは現在の状態のままになります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	X	-	0

例



次のどちらかの状態になると、出力 Q4.0 の信号状態は"0"にリセットされます。

入力 I0.0 と I0.1 の信号状態が"1"の場合

入力 I0.2 の信号状態が"0"の場合

RLO が"0"のとき、出力 Q4.0 の信号状態は変更されません。

タイマ T1 の信号状態は、次の場合にリセットされます。

入力 I0.3 の信号状態が"1"の場合

カウンタ C1 の信号状態は、次の場合にリセットされます。

入力 I0.4 の信号状態が"1"の場合

例に示す回路がアクティブな MCR ゾーン内にある場合、以下が実行されます。

MCR がオンの場合、前述のように、Q4.0、T1、および C1 がリセットされます。

MCR がオフの場合、RLO の状態(パワーフローの状態)に関係なく、Q4.0、T1、および C1 は変更されません。

1.9 ---(S) セットコイル

シンボル

<address>

---(S)

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	セットビット

説明

---(S) (セットコイル)は、前に実行された命令の RLO が"1"の場合に(コイルへのパワーフロー)実行されます。RLO が"1"の場合、エレメントに対し指定された**<address>**は"1"に設定されます。

RLO が 0 の場合は変化がなく、エレメントに対し指定されたアドレスの状態は変わりません。

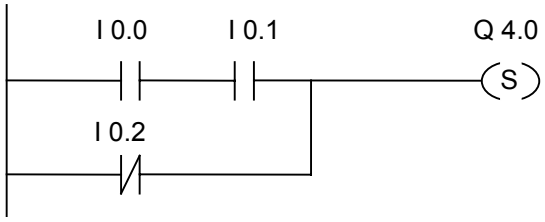
MCR (マスタコントロールリレー)の依存性

MCRの依存性は、アクティブなMCRゾーンにセットコイルが配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになり、かつセットコイルに電流が流れると、アドレス指定されたビットは"1"に設定されます。MCR がオフの場合、パワーフローの状態に関係なく、エレメントに対し指定されたアドレスは現在の状態のままになります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	X	-	0

例



次のどちらかの状態になると、出力 Q4.0 の信号状態は"1"になります。

入力 I0.0 と I0.1 の信号状態が"1"の場合

入力 I0.2 の信号状態が"0"の場合

RLO が"0"のとき、出力 Q4.0 の信号状態は変更されません。

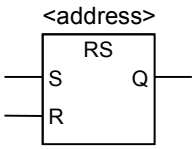
例に示す回路がアクティブな MCR ゾーン内にある場合、以下が実行されます。

MCR がオンの場合、前述のように、Q4.0 が設定されます。

MCR がオフの場合、RLO の状態(パワーフローの状態)に関係なく、Q4.0 は変更されません。

1.10 RS リセット-セットフリップフロップ

シンボル



パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	セットビットまたはリセットビット
S	BOOL	I、Q、M、L、D	イネーブルリセット命令
R	BOOL	I、Q、M、L、D	イネーブルリセット命令
Q	BOOL	I、Q、M、L、D	<address>の信号状態

説明

RS (リセット-セットフリップフロップ)は、R 入力の信号状態が"1"で、S 入力が"0"の場合にリセットされます。一方、R 入力の信号状態が"0"で、S 入力が"1"の場合、フリップフロップは設定されます。S 入力と R 入力のどちらも"1"の場合は、順序が重要になります。SR フリップフロップは、指定された**<address>**でまず、リセット命令を実行し、次にセット命令を実行して、プログラムスキャンを継続できるようにセットされた状態にしておきます。

S(セット)および R(リセット)命令は、RLO が"1"の場合にのみ実行されます。RLO が"0"の場合、これらの命令に対して効果はなく、命令で指定されているアドレスは変わりません。

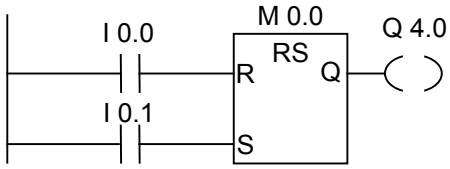
MCR (マスタコントロールリレー)の依存性

MCR の依存性は、アクティブな MCR ゾーンに RS フリップフロップを配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになると、アドレス指定されたビットが"0"にリセットされるか、"1"に設定されます。MCR がオフの場合は、入力の状態に関係なく、指定されたアドレスの現在の状態は変更されません。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



入力 I0.0 の信号状態が"1"、I0.1 の信号状態が"0"の場合、メモリビット M0.0 は設定され、出力 Q4.0 は"0"になります。そうではなく、入力 I0.0 の信号状態が"0"、I0.1 の信号状態が"1"の場合、メモリビット M0.0 はリセットされ、出力 Q4.0 は"1"になります。両方の信号状態が"0"の場合、何も起こりません。両方の信号が"1"の場合は、順序に従ってセット命令が実行されるため、M0.0 が設定され、Q4.0 が"1"になります。

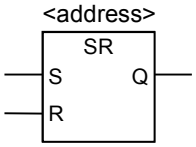
上記の例がアクティブな MCR ゾーンにあると、以下が実行されます。

MCR がオンの場合、前述のように、Q4.0 がリセットされるか、設定されます。

MCR がオフの場合、入力の状態に関係なく、Q4.0 は変更されません。

1.11 SR セット-リセットフリップフロップ

シンボル



パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	セットビットまたはリセットビット
S	BOOL	I、Q、M、L、D	イネーブルセット命令
R	BOOL	I、Q、M、L、D	イネーブルリセット命令
Q	BOOL	I、Q、M、L、D	<address>の信号状態

説明

SR (セット-リセットフリップフロップ)は、S 入力の信号状態が"1"で、R 入力が"0"の場合に設定されます。一方、S 入力の信号状態が"0"で、R 入力が"1"の場合、フリップフロップはリセットされます。S 入力と R 入力のどちらも"1"の場合は、順序が重要になります。SR フリップフロップは、指定された<address>でまず、セット命令を実行し、次にリセット命令を実行して、プログラムスキャンを継続できるようにリセット状態にしておきます。

S(セット)および R(リセット)命令は、RLO が"1"の場合にのみ実行されます。RLO が"0"の場合、これらの命令に対して効果はなく、命令で指定されているアドレスは変わりません。

MCR (マスタコントロールリレー)の依存性

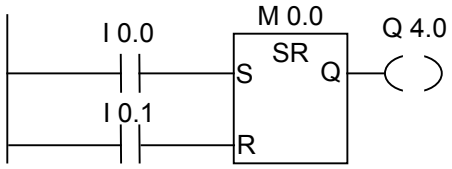
MCR の依存性は、アクティブな MCR ゾーンに SP フリップフロップを配置すると有効になります。アクティブな MCR ゾーンでは、MCR がオンになると、アドレス指定されたビットが"1"に設定さ

れるか、"0"にリセットされます。MCR がオフの場合は、入力の状態に関係なく、指定されたアドレスの現在の状態は変更されません。

ステータスワード

	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



入力 I0.0 の信号状態が"1"、I0.1 の信号状態が"0"の場合、メモリビット M0.0 は設定され、出力 Q4.0 は"1"になります。そうではなく、入力 I0.0 の信号状態が"0"、I0.1 の信号状態が"1"の場合、メモリビット M0.0 はリセットされ、出力 Q4.0 は"0"になります。両方の信号状態が"0"の場合、何も起こりません。両方の信号が"1"の場合は、順序に従ってリセット命令が実行されるため、M0.0 がリセットされ、Q4.0 が"0"になります。

上記の例がアクティブな MCR ゾーンにあると、以下が実行されます。

MCR がオンの場合、前述のように、Q4.0 が設定されるか、リセットされます。

MCR がオフの場合、入力の状態に関係なく、Q4.0 は変更されません。

1.12 ---(N)--- 立ち下がり RLO 信号検出

シンボル

<address>

---(N)

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	信号メモリビット。RLO の前回の信号状態を保存する。

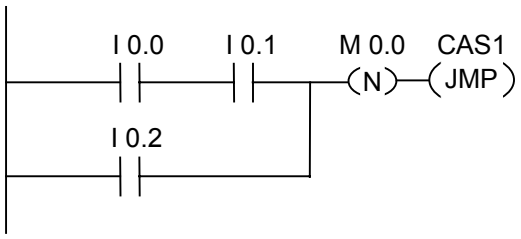
説明

---(N)--- (立ち下がり RLO 信号検出)は、アドレスの信号状態が"1"から"0"に変化するのを検出し、命令の実行後に検出結果として RLO = "1"を表示します。RLO の現在の信号状態と、アドレスの信号メモリビットの信号状態との比較が行われます。アドレスの信号状態が"1"で、かつ命令の実行前に RLO が"0"になった場合、命令の実行後に RLO は"1"(パルス)になります。これ以外の場合、RLO は"0"になります。命令実行前の RLO は、アドレスに保存されます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	x	x	1

例



信号メモリビット M0.0 は、RLO の前の状態を保存します。RLO の信号状態が"1"から"0"に変わると、プログラムはラベル CAS1 へジャンプします。

1.13 ---(P)--- 立ち上がり RLO 信号検出

シンボル

<address>

---(P)---

パラメータ	データタイプ	メモリ領域	説明
<address>	BOOL	I、Q、M、L、D	信号メモリビット。RLO の前回の信号状態を保存する。

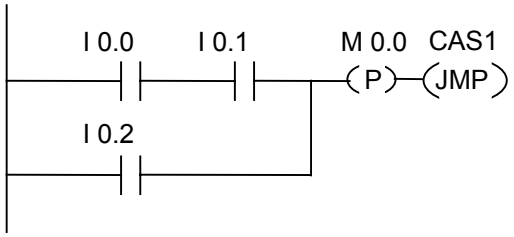
説明

---(P)--- (立ち上がり RLO 信号検出)は、アドレスの信号状態が"0"から"1"に変化するのを検出し、命令の実行後に検出結果として RLO = "1"を表示します。RLO の現在の信号状態と、アドレスの信号メモリビットの信号状態との比較が行われます。アドレスの信号状態が"0"で、かつ命令の実行前に RLO が"1"になった場合、命令の実行後に RLO は"1"(パルス)になります。これ以外の場合、RLO は"0"になります。命令実行前の RLO は、アドレスに保存されます。

ステータスワード

	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	X	X	1

例



信号メモリビット M0.0 は、RLO の前の状態を保存します。RLO の信号状態が"0"から"1"に変わると、プログラムはラベル CAS1 へジャンプします。

1.14 ---(SAVE) RLO を BR メモリに保存

シンボル

---(SAVE)

説明

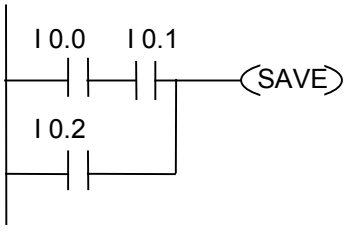
---(SAVE) (RLO を BR メモリに保存)は、ステータスワードの BR ビットに RLO を保存します。最初のチェックビット/FC はリセットされません。このため、BR ビットの状態は、次の回路網の AND 論理演算に含まれます。

"SAVE"命令(LAD、FBD、STL)については、マニュアルやオンラインヘルプで規定されている推奨の使用法ではなく、次のことが適用されます。
SAVE を使用した後に、同じブロックまたは従属ブロック内で BR ビットをチェックすることはお勧めできません。これは、中間で発生する多くの命令によって、BR ビットが修正される可能性があるためです。ブロックを終了するときには、その前に SAVE 命令を使用してください。これは、ENO 出力(= BR ビット)が、RLO ビットの値に設定されるため、ブロックにエラーがあるかどうかチェックできるからです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	X	-	-	-	-	-	-	-	-

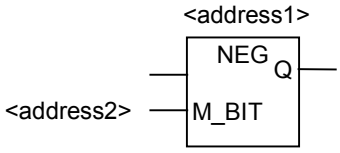
例



回路の状態(=RLO)が、BR ビットに保存されます。

1.15 NEG アドレス立ち下がりパルス

シンボル



パラメータ	データタイプ	メモリ領域	説明
<address1>	BOOL	I、Q、M、L、D	スキャン済みの信号
<address2>	BOOL	I、Q、M、L、D	M_BIT 信号メモリビット。<address1> の前の信号状態を保存する。
Q	BOOL	I、Q、M、L、D	1 回限りの出力

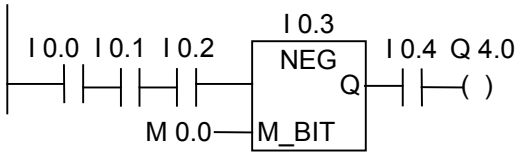
説明

NEG (アドレス立ち下がり信号検出)は、**<address1>** の信号状態を**<address2>**に保存されている前回のスキャン後の信号状態と比較します。RLO の現在の状態が"1"で、前回の状態が"0"の場合(立ち上がり信号の検出)、この命令の実行後、RLO ビットは"1"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	1	x	1

例

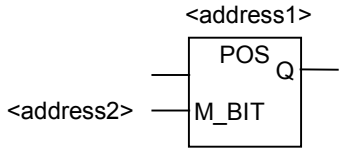


次のいずれかの状態になると、出力 Q4.0 の信号状態は"1"になります。

- 入力 I0.0、I0.1、I0.2 の信号状態が"1"の場合
- 入力 I0.3 が信号立ち下がりの場合
- 入力 I0.4 の信号状態が"1"の場合

1.16 POS アドレス立ち上がり信号検出

シンボル



パラメータ	データタイプ	メモリ領域	説明
<address1>	BOOL	I、Q、M、L、D	スキャン済みの信号
<address2>	BOOL	I、Q、M、L、D	M_BIT 信号メモリビット。<address1> の前の信号状態を保存する。
Q	BOOL	I、Q、M、L、D	1 回限りの出力

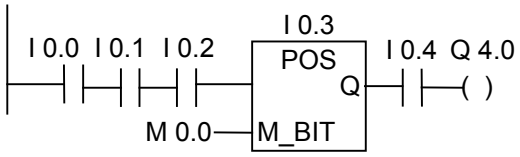
説明

POS (アドレス立ち上がり信号検出)は、**<address1>** の信号状態を**<address2>**に保存されている前回のスキャン後の信号状態と比較します。RLO の現在の状態が"1"で、前回の状態が"0"の場合(立ち上がり信号の検出)、この命令の実行後、RLO ビットは"1"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	1	x	1

例



次のいずれかの状態になると、出力 Q4.0 の信号状態は"1"になります。

- 入力 I0.0、I0.1、I0.2 の信号状態が"1"の場合
- 入力 I0.3 が信号立ち上がりの場合
- 入力 I0.4 の信号状態が"1"の場合

1.17 即時読み取り

説明

即時読み取りファンクションには、次の例のようなネットワークを構成する必要があります。

スピードが要求される用途においては、デジタル入力の現在の状態の読み取り頻度を OB1 スキャンサイクル当たり 1 回という通常の頻度よりも増やすことができます。即時読み取りでは、即時読み取り回路のスキャン時に、入力モジュールからのデジタル入力の状態を読み取ります。この方法を使用しない場合、ユーザーは、次の OB1 スキャンサイクルの最後で、I メモリ領域が P メモリ状態で更新されるのを待たなければなりません。

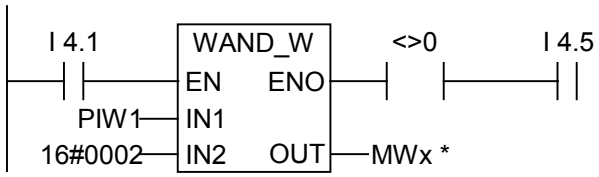
入力モジュールからの入力を即座に読み取るには、入力(I)メモリ領域の代わりに、周辺入力(PI)メモリ領域を使用します。周辺入力メモリ領域は、バイト、ワード、またはダブルワードとして読み取ることができます。したがって、単一のデジタル入力を接点(ビット)エレメントを介して読み取ることはできません。

即時入力の状態に合わせて電圧を通すには、次の方法が実行されます。

1. CPU が、対象となる入力データが格納された PI メモリのワードを読み取ります。
2. 入力ビットがオン("1")の場合にゼロ以外の値が生成された場合、この値をもつ定数と PI メモリのワードとの論理積が求められます。
3. アキュムレータに対し、ゼロ以外の状態かどうかを確認するテストが実行されます。

例

周辺入力 I1.1 の即時読み取りが設定されているラダー回路



***** ネットワークを格納できるようにするには MWx を指定する必要があります。x には、許容される任意の数値を指定できます。

WAND_W 命令の説明

PIW1	0000000000101010
W#16#0002	0000000000000010
結果	0000000000000010

この例では、即時入力 I1.1 は、I4.1 および I4.5 に直列接続されています。

ワード PIW1 には、I1.1 の直接のステータスが格納されます。PIW1 は、W#16#0002 と AND されます。PB1 の I1.1(2 番目のビット)が True("1")の場合、結果は 0 と等しくなりません。WAND_W 命令の結果が 0 以外の場合、接点 A<>0 は電圧を通します。

1.18 即時書き込み

説明

即時書き込みファンクションでは、次の例のような回路を構成する必要があります。

スピードが要求される用途においては、デジタル出力の現在の状態を出力モジュールへ送る速度をOB1 スキャンサイクルの終了時に 1 回という通常の速度よりも速くする必要があります。即時書き込みでは、即時書き込み回路のスキャン時に、デジタル出力を入力モジュールに書き込みます。この方法を使用しない場合、ユーザーは、次の OB1 スキャンサイクルの最後で、Q メモリ領域が P メモリ状態で更新されるのを待たなければなりません。

出力を出力モジュールに即座に書き込むには、出力(Q)メモリ領域の代わりに、周辺出力(PQ)メモリ領域を使用します。周辺出力メモリ領域は、バイト、ワード、またはダブルワードとして読み取ることができます。したがって、単一のデジタル出力をコイルエレメントを介して更新することはできません。デジタル出力の状態を出力モジュールに即座に書き込むには、対応する PQ メモリ(直接出力モジュールアドレス)に、関連するビットをもつ Q メモリのバイト、ワード、またはダブルワードを条件付きでコピーします。

注意

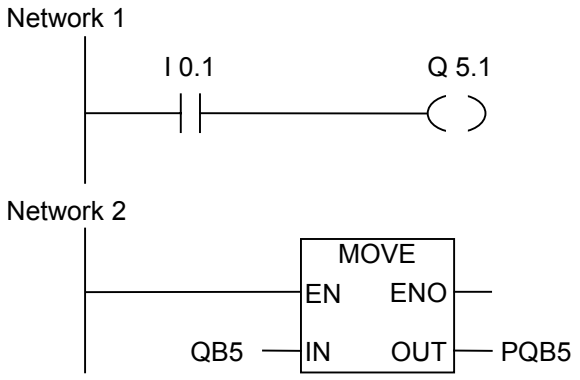
- Q メモリ全体のバイトが出力モジュールに書き込まれるため、そのバイトの全出力ビットは、即時出力の実行時に更新されます。
- 出力モジュールへ送信する必要のないプログラムで出力ビットが中間状態(1/0)になると、即時書き込みにより、危険な状態(出力における一時的パルス)が発生する場合があります。
- 設計上の一般的なルールでは、外部出力モジュールがプログラム内でコイルとして参照されるのは一度に留めます。このようにすれば、即時出力の問題のほとんどは回避できます。

例

周辺デジタル出力モジュール 5 への即時書き込みのラダー回路は、チャネル 1 に相当します。

アドレス指定された出力 Q バイト(QB5)のビット状態は、変更済みまたは未変更のどちらかです。Q5.1 は、ネットワーク 1 の I0.1 の信号状態に割り付けられます。QB5 は、対応する直接周辺出力のメモリ領域(PQB5)にコピーされます。

ワード PIW1 には、I1.1 の直接のステータスが格納されます。PIW1 は、W#16#0002 と AND されます。PB1 の I1.1(2 番目のビット)が True("1")の場合、結果は 0 と等しくなりません。WAND_W 命令の結果が 0 以外の場合、接点 A<>0 は電圧を通します。



上記の例では、Q5.1 には望ましい即時出力ビットが示されます。
バイト PQB5 には、ビット Q5.1 の即時出力ステータスが示されます。
PQB5 の他の 7 ビットも、MOVE(コピー)命令によって更新されます。

2 比較命令

2.1 比較命令の概要

説明

IN1 と IN2 は、選択した比較タイプに従って比較されます。

== IN1 は IN2 と等しい
<> IN1 は IN2 と等しくない
> IN1 は IN2 より大きい
< IN1 は IN2 より小さい
>= IN1 は IN2 以上
<= IN1 は IN2 以下

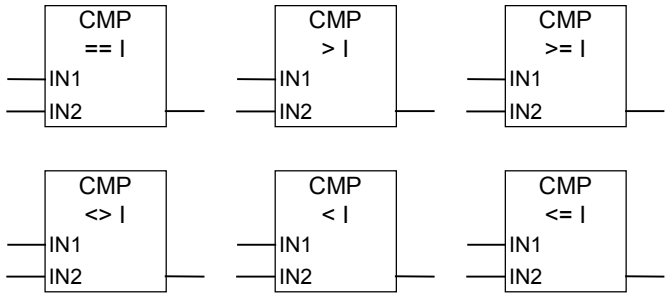
比較が真の場合、ファンクションの RLO は"1"になります。比較エレメントを直列で使用した場合、回路ネットワークの RLO に AND でリンクされます。ボックスを並列で使用した場合、回路ネットワークの RLO に OR でリンクされます。

使用可能な比較命令を次に示します。

- CMP ? I 整数の比較
- CMP ? D 倍長整数の比較
- CMP ? R 実数の比較

2.2 CMP ? I 整数の比較

シンボル



パラメータ	データタイプ	メモリ領域	説明
box input	BOOL	I、Q、M、L、D	前回の論理演算の結果
box output	BOOL	I、Q、M、L、D	比較の結果。box input の RLO が 1 の場合、さらに処理を実行できる。
IN1	INT	I、Q、M、L、D または接点	比較する最初の値
IN2	INT	I、Q、M、L、D または接点	比較する 2 番目の値

説明

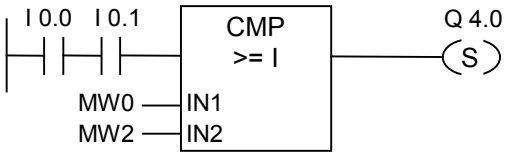
CMP ? I (整数の比較)は、通常接点のように使用できます。通常接点を配置できる場所ならどこにでも配置できます。IN1 と IN2 の比較は、選択した比較のタイプに従って実行されます。

比較が真の場合、ファンクションの RLO は"1"になります。ボックスを直列で使用した場合、回路全体の RLO に AND でリンクされます。ボックスを並列で使用した場合、回路ネットワークの RLO に OR でリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	0	-	0	x	x	1

例

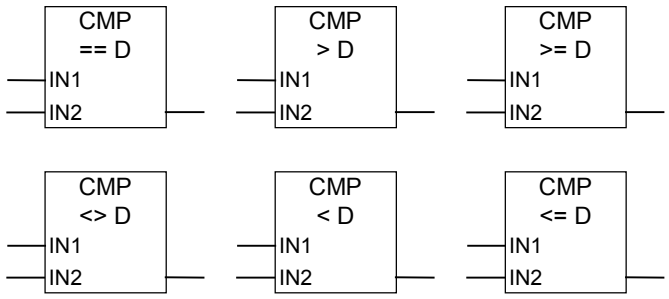


次の条件があてはまる場合、出力 Q4.0 が設定されます。

- 入力 I0.0 と I0.1 の信号状態が"1"
- AND MW0 >= MW2

2.3 CMP ? D 倍長整数の比較

シンボル



パラメータ	データタイプ	メモリ領域	説明
box input	BOOL	I、Q、M、L、D	前回の論理演算の結果
box output	BOOL	I、Q、M、L、D	比較の結果。box input の RLO が 1 の場合、さらに処理を実行できる。
IN1	DINT	I、Q、M、L、D または接点	比較する最初の値
IN2	DINT	I、Q、M、L、D または接点	比較する 2 番目の値

説明

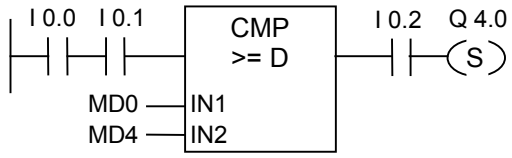
CMP ? D (倍長整数の比較)は、通常接点のように使用できます。通常接点を配置できる場所ならどこにでも配置できます。IN1 と IN2 の比較は、選択した比較のタイプに従って実行されます。

比較が真の場合、ファンクションの RLO は"1"になります。比較エレメントを直列で使用した場合、回路ネットワークの RLO に AND でリンクされます。ボックスを並列で使用した場合、回路ネットワークの RLO に OR でリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	0	-	0	x	x	1

例

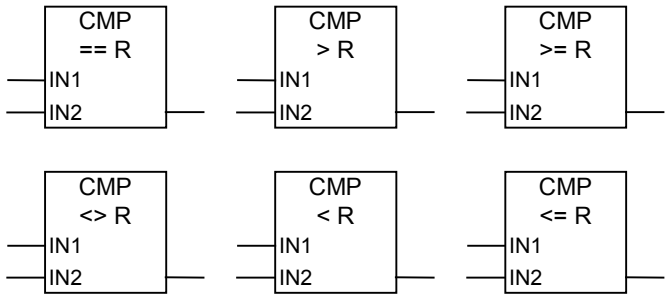


次の条件があてはまる場合、出力 Q4.0 が設定されます。

- 入力 I0.0 と I0.1 の信号状態が"1"
- MD0 >= MD4
- 入力 I0.2 の信号状態が"1"

2.4 CMP ? R 実数の比較

シンボル



パラメータ	データタイプ	メモリ領域	説明
box input	BOOL	I、Q、M、L、D	前回の論理演算の結果
box output	BOOL	I、Q、M、L、D	比較の結果。box input の RLO が 1 の場合、さらに処理を実行できる。
IN1	REAL	I、Q、M、L、D または接点	比較する最初の値
IN2	REAL	I、Q、M、L、D または接点	比較する 2 番目の値

説明

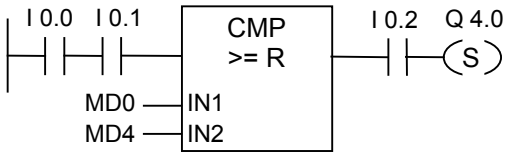
CMP ? R (実数の比較)は、通常接点のように使用できます。通常接点を配置できる場所ならどこにでも配置できます。IN1 と IN2 の比較は、選択した比較のタイプに従って実行されます。

比較が真の場合、ファンクションの RLO は"1"になります。ボックスを直列で使用した場合、回路全体の RLO に AND でリンクされます。ボックスを並列で使用した場合、回路ネットワークの RLO に OR でリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

例



次の条件があてはまる場合、出力 Q4.0 が設定されます。

- 入力 I0.0 と I0.1 の信号状態が"1"
- MD0 >= MD4
- 入力 I0.2 の信号状態が"1"

3 変換命令

3.1 変換命令の概要

説明

変換命令はパラメータ IN の内容を読み取り、これらを変換するか符号を変更します。この結果は、パラメータ OUT で照会できます。

使用可能な変換命令を次に示します。

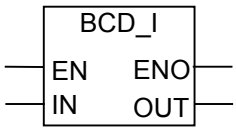
- BCD_I　BCD から整数へ
- I_BCD　整数から BCD へ
- BCD_DI　BCD から倍長整数へ
- I_DINT　整数から倍長整数へ
- DI_BCD　倍長整数から BCD へ
- DI_REAL　倍長整数から浮動小数点へ

- INV_I　整数の 1 の補数
- INV_DI　倍長整数の 1 の補数
- NEG_I　整数の 2 の補数
- NEG_DI　倍長整数の 2 の補数

- NEG_R　負の浮動小数点数
- ROUND　倍長整数への丸め
- TRUNC　倍長整数部分の切り捨て
- CEIL　切り上げ
- FLOOR　切り下げ

3.2 BCD_I BCD から整数へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	WORD	I、Q、M、L、D	BCD 番号
OUT	INT	I、Q、M、L、D	BCD 番号の整数値

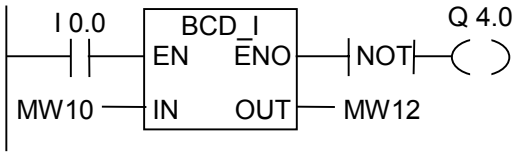
説明

BCD_I(BCD から整数へ変換)は、IN パラメータの内容を 3 桁の BCD コード番号(+/- 999)として読み取り、それを整数値(16 ビット)に変換します。この整数は、パラメータ OUT で出力されます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

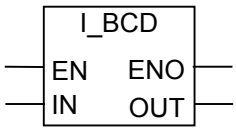
例



I0.0 が"1"のとき、MW10 の内容が、3 桁の BCD コード番号として読み取られ、整数に変換されます。この結果は、MW12 に格納されます。変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.3 I_BCD 整数から BCD へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	INT	I、Q、M、L、D	整数番号
OUT	WORD	I、Q、M、L、D	整数番号の BCD 値

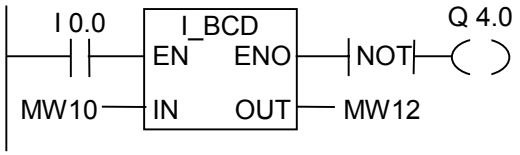
説明

I_BCD (整数から BCD へ変換)は、IN パラメータの内容を整数値(16 ビット)として読み取り、それを 3 桁の BCD コード番号(+/- 999)に変換します。この結果は、パラメータ OUT で出力されます。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	x	x	0	x	x	1

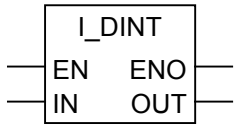
例



I0.0 が"1"のとき、MW10 の内容が、整数として読み取られ、3 桁の BCD コード番号に変換されます。この結果は、MW12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

3.4 I_DINT 整数から倍長整数へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	INT	I、Q、M、L、D	変換対象の整数値
OUT	DINT	I、Q、M、L、D	結果(倍長整数)

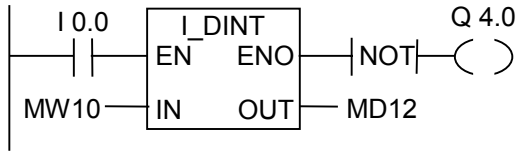
説明

I_DINT (整数から倍長整数へ変換)は、IN パラメータの内容を整数(16 ビット)として読み取り、それを倍長整数(32 ビット)に変換します。この結果は、パラメータ OUT で出力されます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

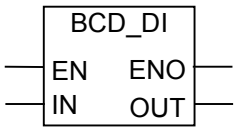
例



I0.0 が"1"の場合、MW10 の内容が、整数値として読み取られ、倍長整数に変換されます。この結果は、MD12 に格納されます。変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.5 BCD_DI BCD から倍長整数へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DWORD	I、Q、M、L、D	BCD 番号
OUT	DINT	I、Q、M、L、D	BCD 番号の倍長整数値

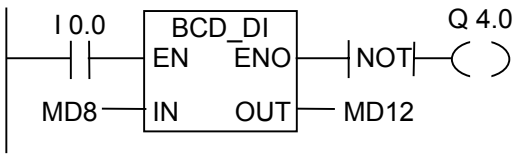
説明

BCD_DI (BCD から倍長整数へ変換)は、IN パラメータの内容を 7 桁の BCD コード番号(+/- 9999999)として読み取り、それを倍長整数値(32 ビット)に変換します。この倍長整数は、パラメータ OUT で出力されます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

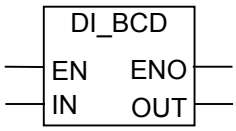
例



I0.0 が"1"のとき、MD8 の内容が、7 桁の BCD コード番号として読み取られ、倍長整数に変換されます。この結果は、MD12 に格納されます。変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.6 DI_BCD 倍長整数から BCD へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DINT	I、Q、M、L、D	倍長整数番号
OUT	DWORD	I、Q、M、L、D	倍長整数番号の BCD 値

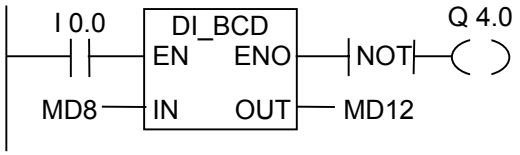
説明

DI_BCD (倍長整数から BCD へ変換)は、IN パラメータの内容を倍長整数(32 ビット)として読み取り、それを 7 桁の BCD コード番号(+/- 9999999)に変換します。この結果は、パラメータ OUT で出力されます。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	x	x	0	x	x	1

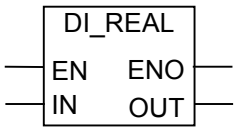
例



I0.0 が"1"のとき、MD8 の内容が、倍長整数として読み取られ、7 桁の BCD 番号に変換されます。この結果は、MD12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

3.7 DI_REAL 倍長整数から浮動小数点へ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DINT	I、Q、M、L、D	変換対象の倍長整数値
OUT	REAL	I、Q、M、L、D	結果(浮動小数点数)

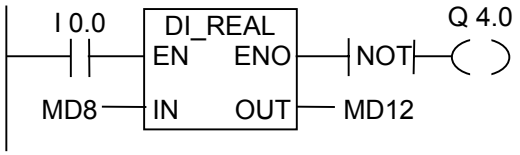
説明

DI_REAL (倍長整数から浮動小数点へ変換)は、IN パラメータの内容を倍長整数として読み取り、それを浮動小数点数に変換します。この結果は、パラメータ OUT で出力されます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

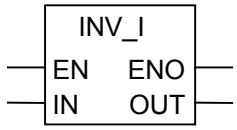
例



I0.0 が"1"のとき、MD8 の内容が、倍長整数として読み取られ、浮動小数点数に変換されます。この結果は、MD12 に格納されます。変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.8 INV_I 整数の 1 の補数

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	INT	I、Q、M、L、D	整数入力値
OUT	INT	I、Q、M、L、D	整数 IN の 1 の補数

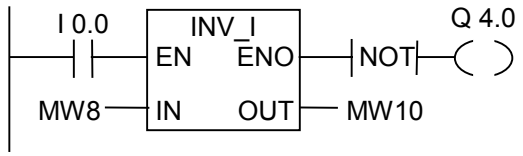
説明

INV_I (整数の 1 の補数)は、IN パラメータの内容を読み取り、16 進数マスク W#16#FFFF を使ってブール XOR ファンクションを実行します。この命令は、すべてのビットを逆の状態に切り替えます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

例



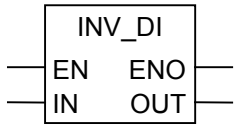
I0.0 が"1"の場合、MW8 のすべてのビットが逆転します。次にその例を示します。

MW8 = 01000001 10000001 の結果は MW10 = 10111110 01111110 になります。

変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.9 INV_DI 倍長整数の 1 の補数

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DINT	I、Q、M、L、D	倍長整数の入力値
OUT	DINT	I、Q、M、L、D	倍長整数 IN の 1 の補数

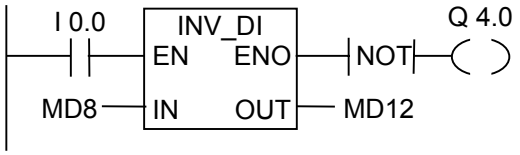
説明

INV_DI (倍長整数の 1 の補数)は、IN パラメータの内容を読み取り、16 進マスク W#16#FFFF FFFF を使ってブール XOR ファンクションを実行します。この命令は、すべての 1 ビットを逆の状態に切り替えます。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

例



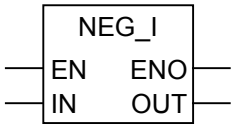
I0.0 が"1"の場合、MW8 のすべてのビットが逆転します。次にその例を示します。

MD8 = F0FF FFF0 は、MD12 = 0F00 000F になります。

変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.10 NEG_I 整数の 2 の補数

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	INT	I、Q、M、L、D	整数入力値
OUT	INT	I、Q、M、L、D	整数 N の 2 の補数

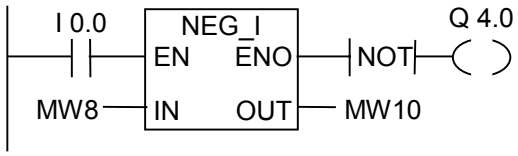
説明

NEG_I (整数の 2 の補数)は、IN パラメータの内容を読み取り、2 の補数命令を実行します。2 の補数命令は(-1)で乗算するのと等しく、符号を変換します(正数から負数への変換など)。ENO と EN の信号状態は常に同じになります。ただし、EN の信号状態が 1 でオーバーフローが発生した場合は、ENO の信号状態が 0 になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

例



I0.0 が"1"の場合、MW8 の値に逆の記号が付き、パラメータ OUT により MW10 に出力されます。

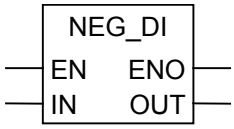
MW8 = + 10 の結果は MW10 = - 10.

変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

EN の信号状態が 1 のときにオーバーフローが発生すると、ENO の信号状態は 0 になります。

3.11 NEG_DI 倍長整数の 2 の補数

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DINT	I、Q、M、L、D	倍長整数の入力値
OUT	DINT	I、Q、M、L、D	IN の値の 2 の補数

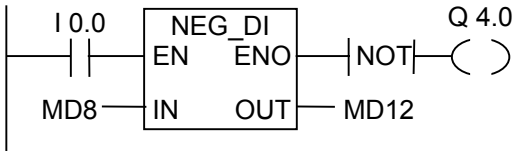
説明

NEG_DI (倍長整数の 2 の補数)は、IN パラメータの内容を読み取り、2 の補数命令を実行します。2 の補数命令は(-1)で乗算するのと等しく、符号を変換します(正数から負数への変換など)。ENO と EN の信号状態は常に同じになります。ただし、EN の信号状態が 1 でオーバーフローが発生した場合は、ENO の信号状態が 0 になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

例



I0.0 が"1"の場合、MD8 の値に逆の記号が付き、パラメータ OUT により MD12 に出力されます。

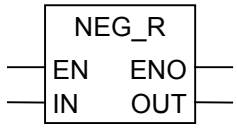
MD8 = + 1000 の結果は MD12 = - 1000.

変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

EN の信号状態が 1 のときにオーバーフローが発生すると、ENO の信号状態は 0 になります。

3.12 NEG_R 負の浮動小数点数

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D	浮動小数点数の入力値
OUT	REAL	I、Q、M、L、D	マイナス記号付きの浮動小数点数 IN

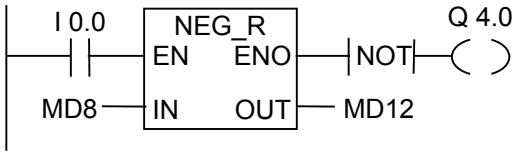
説明

NEG_R (負の浮動小数点)は、IN パラメータの内容を読み取り、記号を変換します。この命令は(-1)で乗算するのと等しく、符号を変換します(正数から負数への変換など)。ENO と EN の信号状態は常に同じです。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	-	-	0	x	x	1

例



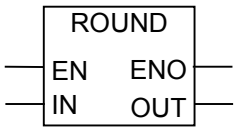
I0.0 が"1"の場合、MD8 の値に逆の記号が付き、パラメータ OUT により MD12 に出力されます。

MD8 = + 6.234 の結果は MD12 = - 6.234.

変換が実行されない場合(ENO = EN = 0)、出力 Q4.0 は"1"になります。

3.13 ROUND 倍長整数への丸め

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D	四捨五入対象の値
OUT	DINT	I、Q、M、L、D	IN の値を直近の整数に四捨五入

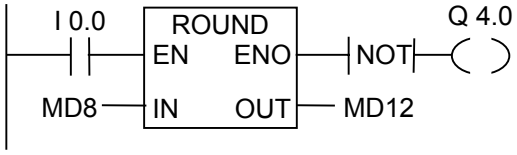
説明

ROUND (倍長整数への丸め)は、IN パラメータの内容を浮動小数点数として読み取り、それを倍長整数(32 ビット)に変換します。この結果、最も近い整数値("近似値への丸め")を得ることができます。浮動小数点数が 2 つの整数値の間にある場合は、偶数が返されます。この結果は、パラメータ OUT で出力されます。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	x	x	0	x	x	1

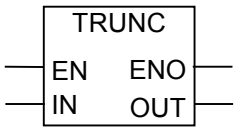
例



I0.0 が"1"のとき、MD8 の内容が、浮動小数点数として読み取られ、もっとも近い倍長整数に変換されます。この[近似値への丸め]ファンクションの結果は、MD12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

3.14 TRUNC 倍長整数部分の切り捨て

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D	変換対象の浮動小数点数
OUT	DINT	I、Q、M、L、D	IN の値の整数部分

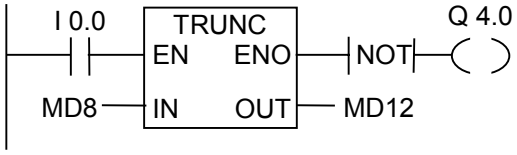
説明

TRUNC (倍長整数の切り捨て))は、IN パラメータの内容を浮動小数点数として読み取り、それを倍長整数(32 ビット)に変換します。この方法は"ゼロモードに丸め"ファンクションに相当し、この結果から得られた倍長整数は、パラメータ OUT で出力されます。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	x	x	0	x	x	1

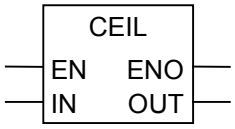
例



I0.0 が"1"のとき、MD8 の内容が、実数として読み取られ、倍長整数に変換されます。浮動小数点数の整数部分が結果となり、MD12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

3.15 CEIL 切り上げ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D	変換対象の浮動小数点数
OUT	DINT	I、Q、M、L、D	より大きい値の中で直近の倍長整数

説明

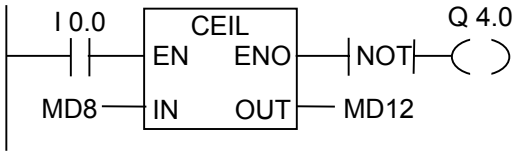
CEIL (切り上げ)は、IN パラメータの内容を浮動小数点数として読み取り、それを倍長整数(32 ビット)に変換します。この浮動小数点数よりも大きく最も小さい整数が結果となります("丸め-無限大")。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容*:	X	-	-	X	X	0	X	X	1
書き込みの内容**:	0	-	-	-	-	0	0	0	1

* ファンクションが実行される(EN = 1)
** ファンクションが実行されない(EN = 0)

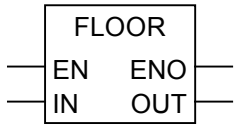
例



I0.0 が 1 の場合は、MD8 の内容が、**Round** ファンクションを使用して倍長整数に変換される浮動小数点数として読み取られます。この結果は、MD12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

3.16 FLOOR 切り下げ

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D	変換対象の浮動小数点数
OUT	DINT	I、Q、M、L、D	より近い値の中で最大の倍長整数

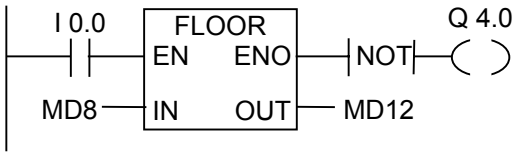
Yn

FLOOR (切り下げ)は、IN パラメータの内容を浮動小数点数として読み取り、それを倍長整数(32 ビット)に変換します。この浮動小数点数よりも小さく最も大きい整数が結果となります("丸め-無限大")。オーバーフローが発生すると、ENO は"0"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	x	x	0	x	x	1

例



I0.0 が"1"の場合、MD8 の内容が浮動小数点数として読み取られ、"丸め-無限大"モードにより倍長整数に変換されます。この結果は、MD12 に格納されます。オーバーフローが発生したか、命令が実行されなかった(I0.0 = 0)場合は、出力 Q4.0 が"1"になります。

4 カウンタ命令

4.1 カウンタ命令の概要

メモリ内の領域

カウンタは、CPU のメモリ内にカウンタ用に確保された領域を持っています。このメモリ領域は、各カウンタアドレスに対して 1 つの 16 ビットワードを確保します。ラダー論理命令群には 256 個のカウンタがあります。

カウンタ命令は、カウンタメモリ領域にアクセスする唯一のファンクションです。

カウンタ値

カウンタワードのビット 0～9 には、カウント値がバイナリコードで格納されます。カウンタが設定されると、カウンタ値はカウンタワードへ転送されます。カウント値の範囲は 0～999 です。

次のカウンタ命令を使用すれば、この範囲内でカウント値を変更することができます。

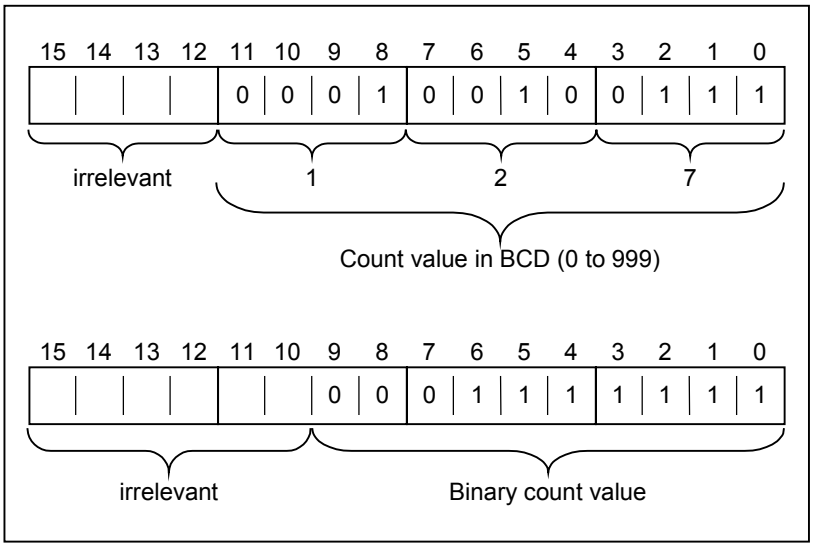
- S_CUD　カウントアップダウン
- S_CD　　カウントダウン
- S_CU　　カウントアップ
- ---(SC)　カウンタコイルの設定
- ---(CU)　カウントアップコイル
- ---(CD)　カウントダウンコイル

カウンタ内のビットコンフィグレーション

0～999 の範囲内の数値を入力することで、カウンタに事前設定値を指定できます。たとえば 127 の場合は、C#127 のように入力します。C#は 2 進化 10 進数形式を表します(BCD 形式: 4 ビットのセットごとに、10 進値の 2 進コードが 1 つ格納されています)。

0 番から 11 番のカウンタビットは、BCD 表記でカウンタ値を表します。

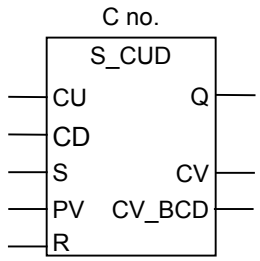
次の図に、カウント値 127 をロードした後のカウンタの内容と、カウンタが設定された後のカウンタセルの内容を示します。



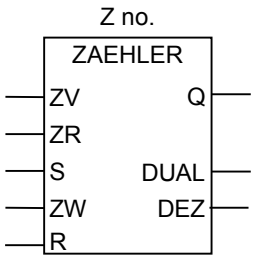
4.2 S_CUD カウントアップダウン

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
C no.	Z no.	COUNTER	C	カウンタ識別番号。範囲は CPU により異なる。
CU	ZV	BOOL	I、Q、M、L、D	カウントアップ入力
CD	ZR	BOOL	I、Q、M、L、D	カウントダウン入力
S	S	BOOL	I、Q、M、L、D	カウンタプリセット用入力
PV	ZW	WORD	I、Q、M、L、D または接点	0～999 の範囲内のカウンタ値を C#<value>として入力します。
PV	ZW	WORD	I、Q、M、L、D	カウンタプリセットの値
R	R	BOOL	I、Q、M、L、D	リセット入力
CV	DUAL	WORD	I、Q、M、L、D	現在のカウンタ値(16 進数)
CV_BCD	DEZ	WORD	I、Q、M、L、D	現在のカウンタ値(BCD 値)
Q	Q	BOOL	I、Q、M、L、D	カウンタのステータス

説明

S_CUD (カウントアップダウン)は、入力 S が信号立ち上がりの場合に、入力 PV の値でプリセットされます。入力 R に"1"が入っている場合、カウンタはリセットされ、カウントは 0 に設定されます。入力 CU の信号状態が"0"から"1"に変わり、カウンタの値が"9990"未満の場合、カウンタは 1 ずつ増えます。入力 CD に立ち上がりエッジがあり、カウンタの値がゼロより大きい場合、カウンタは 1 ずつ減ります。

両方のカウント入力が信号立ち上がりの場合、両方の命令が実行され、カウンタ値は変わりません。

カウンタが設定され、入力 CU/CD が RLO = 1 の場合は、信号立ち上がりから信号立ち下がりへ(またはこの逆)の変更がない場合でも、カウンタ**は次**のスキャンサイクルで 1 度カウントを行います。

カウントが 0 より大きい場合、出力 Q の信号状態は"1"になり、カウントがゼロの場合は"0"になります。

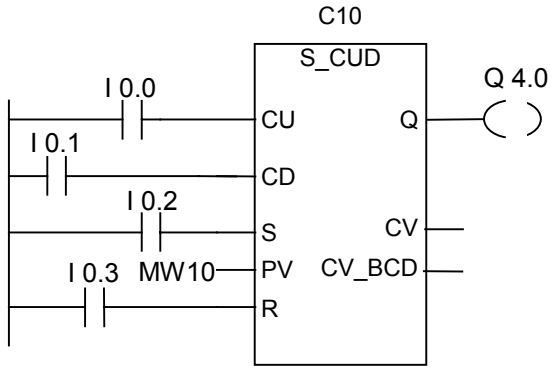
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

注記

複数のプログラムポイントでカウンタを使用しないでください(カウントエラーが発生する危険があります)。

例

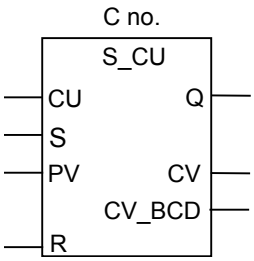


I0.2 が"0"から"1"に変わると、MW10 の値でカウンタが事前設定されます。I0.0 の信号状態が"0"から"1"に変わると、カウンタ C10 の値が 1 増えます。ただし、C10 の値が"999"に等しい場合は例外です。I0.1 が"0"から"1"に変わると、C10 は 1 減ります。ただし、C10 の値が"0"に等しい場合は例外です。C10 が 0 でない場合、Q4.0 は"1"になります。

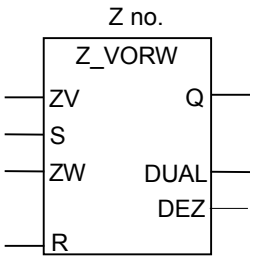
4.3 S_CU カウントアップ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
C no.	Z no.	COUNTER	C	カウンタ識別番号。範囲は CPU により異なる。
CU	ZV	BOOL	I、Q、M、L、D	カウントアップ入力
S	S	BOOL	I、Q、M、L、D	カウンタプリセット用入力
PV	ZW	WORD	I、Q、M、L、D または接点	0～999 の範囲内のカウンタ値を C#<value>として入力します。
PV	ZW	WORD	I、Q、M、L、D	カウンタプリセットの値
R	R	BOOL	I、Q、M、L、D	リセット入力
CV	DUAL	WORD	I、Q、M、L、D	現在のカウンタ値(16 進数)
CV_BCD	DEZ	WORD	I、Q、M、L、D	現在のカウンタ値(BCD 値)
Q	Q	BOOL	I、Q、M、L、D	カウンタのステータス

説明

S_CU (カウントアップ)は、入力 S が信号立ち上がりの場合に、入力 PV の値でプリセットされます。

入力 R に"1"が入っている場合、カウンタはリセットされ、カウントはゼロに設定されます。

入力 CU の信号状態が"0"から"1"に変わり、カウンタの値が"999"未満の場合、カウンタは 1 ずつ増えます。

カウンタが設定され、入力 CU が RLO = 1 の場合は、信号立ち上がりから信号立ち下がりへ(またはこの逆)の変更がない場合でも、カウンタ**は次**のスキャンサイクルで 1 度カウントを行います。

カウントが 0 より大きい場合、出力 Q の信号状態は"1"になり、カウントがゼロの場合は"0"になります。

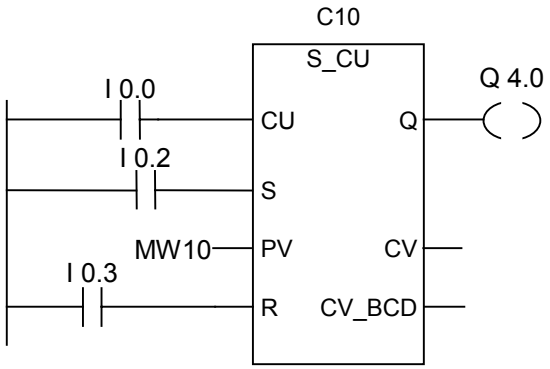
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

注記

複数のプログラムポイントでカウンタを使用しないでください(カウントエラーが発生する危険があります)。

例

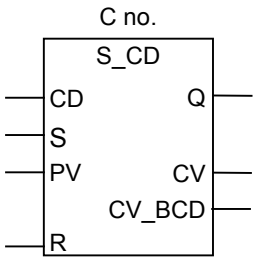


I0.2 が"0"から"1"に変わると、MW10 の値でカウンタが事前設定されます。I0.0 の信号状態が 0 から 1 に変わると、カウンタ C10 の値が 1 ずつ増えます。ただし、C10 の値が"999"のときは例外です。C10 が 0 でない場合、Q4.0 は"1"になります。

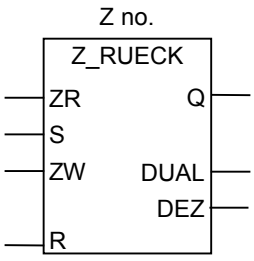
4.4 S_CD カウントダウン

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
C no.	Z no.	COUNTER	C	カウンタ識別番号。範囲は CPU により異なる。
CD	ZR	BOOL	I、Q、M、L、D	カウントダウン入力
S	S	BOOL	I、Q、M、L、D	カウンタプリセット用入力
PV	ZW	WORD	I、Q、M、L、D または接点	0～999 の範囲内のカウンタ値を C#<value>として入力します。
PV	ZW	WORD	I、Q、M、L、D	カウンタプリセットの値
R	R	BOOL	I、Q、M、L、D	リセット入力
CV	DUAL	WORD	I、Q、M、L、D	現在のカウンタ値(16 進数)
CV_BCD	DEZ	WORD	I、Q、M、L、D	現在のカウンタ値(BCD 値)
Q	Q	BOOL	I、Q、M、L、D	カウンタの状態

説明

S_CD (カウントダウン命令)は、入力 S が信号立ち上がりの場合に、入力 PV の値で設定されます。

入力 R に"1"が入っている場合、カウンタはリセットされ、カウントはゼロに設定されます。

入力 CD の信号状態が"0"から"1"に変わり、カウンタの値がゼロより大きい場合、カウンタは 1 ずつ減ります。

カウンタが設定され、入力 CD が RLO = 1 の場合は、信号立ち上がりから信号立ち下がりへ(またはこの逆)の変更がない場合でも、カウンタ**は次**のスキャンサイクルで 1 度カウントを行います。

カウントが 0 より大きい場合、出力 Q の信号状態は"1"になり、カウントがゼロの場合は"0"になります。

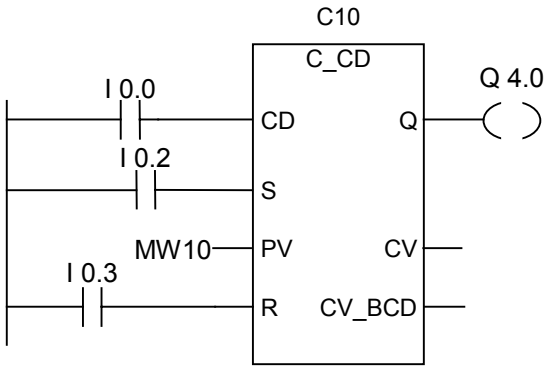
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

注記

複数のプログラムポイントでカウンタを使用しないでください(カウントエラーが発生する危険があります)。

例



I0.2 が"0"から"1"に変わると、MW10 の値でカウンタが事前設定されます。I0.0 の信号状態が 0 から 1 に変わると、カウンタ C10 の値が 1 ずつ減ります。ただし、C10 の値が"0"のときは例外です。C10 が 0 でない場合、Q4.0 は"1"になります。

4.5 ---(SC) カウンタ値の設定

シンボル

英語	ドイツ語
<C no.>	<Z no.>
---(SC)	**---(SZ)**
<preset value>	<preset value>

パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
<C no.>	<Z no.>	COUNTER	C	設定対象のカウンタ番号
<preset value>	<preset value>	WORD	I、Q、M、L、D または接点	設定する BCD の値 (0～999)

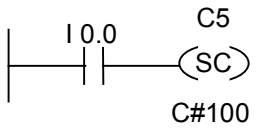
説明

---(SC)(カウンタ値のセット)は、RLO が信号立ち上がりのときに実行します。このとき、事前設定値が、指定されたカウンタに転送されます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	-	-	-	-	0	x	-	0

例



入力 I0.0 が信号立ち上がりのとき("0"から"1"に変わる)、カウンタ C5 は 100 で設定されます。信号立ち上がりでない場合、カウンタ C5 の値は変更されません。

4.6 ---(CU) カウントアップコイル

シンボル

英語	ドイツ語
<C no.>	<Z no.>
---(CU)	**---(ZV)**

パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
<C no.>	<Z no.>	COUNTER	C	カウンタ識別番号。範囲は CPU により異なる。

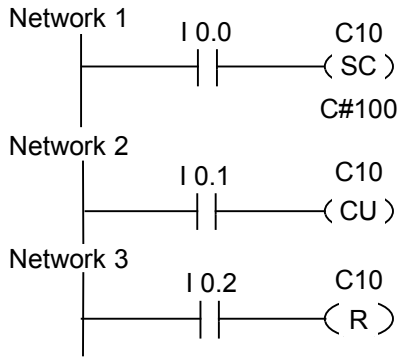
説明

---(CU)(カウントアップコイル)命令は、RLO に立ち上がりエッジがあり、カウンタ値が"999"より小さいときに、指定したカウンタの値を 1 増やします。RLO に立ち上がりエッジがないか、カウンタの値がすでに"999"である場合、カウンタの値は変わりません。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



入力 I0.0 の信号状態が"0"から"1"に変わると(RLO が信号立ち上がり)、事前設定値 100 がカウンタ C10 にロードされます。

入力 I0.1 の信号状態が"0"から"1"に変わると(RLO に立ち上がりエッジがある)、C10 の値が"999"でない限り、カウンタ C10 のカウンタ値は 1 ずつ増えます。RLO に立ち上がりエッジがない場合、カウンタの値は変わりません。

10.2 の信号状態が"1"の場合、カウンタ C10 は"0"にリセットされます。

4.7 ---(CD) カウントダウンコイル

シンボル

英語	ドイツ語
<C no.>	<Z no.>
---(CD)	**---(ZD)**

パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
<C no.>	<Z no.>	COUNTER	C	カウンタ識別番号。範囲は CPU により異なる。

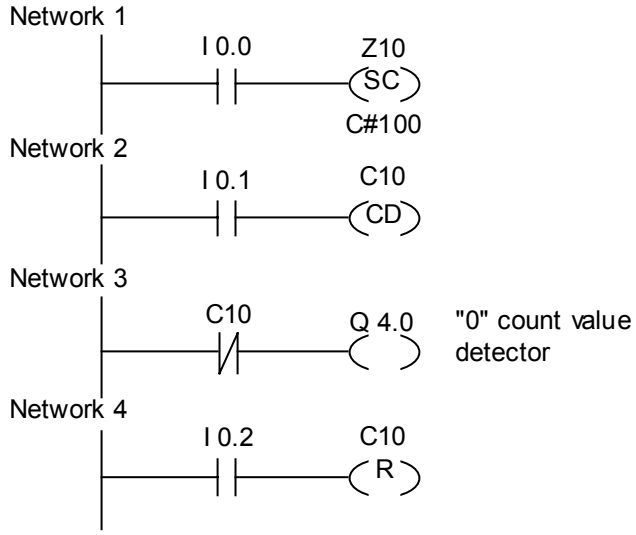
説明

---(CD) (カウントダウンコイル)命令は、RLO に立ち上がりエッジがあり、カウンタ値が"0"より大きいときに、指定したカウンタの値を 1 増やします。RLO に立ち上がりエッジがないか、カウンタの値がすでに"0"である場合、カウンタの値は変わりません。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



入力 I0.0 の信号状態が"0"から"1"に変わると(RLO が信号立ち上がり)、事前設定値 100 がカウンタ C10 にロードされます。

4.7 ---(CD) カウントダウンコイル

入力 I0.1 の信号状態が"0"から"1"に変わると(RLO に立ち上がりエッジがある)、C10 の値が"0"でない限り、カウンタ C10 のカウンタ値は 1 ずつ減ります。RLO に立ち上がりエッジがない場合、カウンタの値は変わりません。

カウンタ値が 0 の場合、Q4.0 はオンになります。

入力 I0.2 の信号状態が"1"の場合、カウンタ C10 は"0"にリセットされます。

5 データブロック命令

5.1 ---(OPN) データブロックを開く: DB または DI

シンボル

<DB 番号>または<DI 番号>

---(OPN)

パラメータ	データタイプ	メモリ領域	説明
<DB no.> <DI no.>	BLOCK_DB	DB, DI	DB/DI の番号; 範囲は CPU により異なる

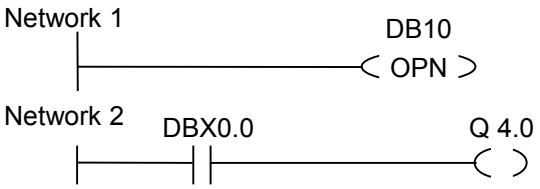
説明

---(OPN) (データブロックを開く)は、共有データブロック(DB)またはインスタンスデータブロック(DI)を開きます。このファンクションは、データブロックの条件なし呼び出しです。データブロックの番号は、DB レジスタまたは DI レジスタに転送されます。その後に実行される DB コマンドと DI コマンドは、レジスタの内容に応じて、対応するブロックにアクセスします。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	-	-	-	-

例



データブロック 10(DB10)が開きます。接点アドレス(DBX0.0)は、DB10 に格納された現行データレコードのデータバイト 0 のビット 0 を参照します。このビットの信号状態は、出力 Q 4.0 に割り付けられます。

5.1 ---(OPN) データブロックを開く: DB または DI

6 ロジックコントロール命令

6.1 論理制御命令の概要

説明

論理制御命令は、オーガニゼーションブロック(OB)、ファンクションブロック(FB)、ファンクション(FC)といったすべての論理ブロックで使用できます。

次のファンクションを実行する論理制御命令があります。

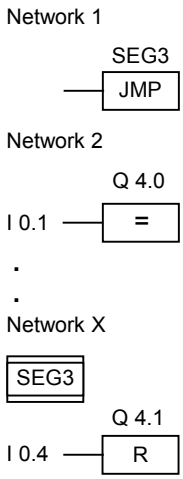
- ---(JMP)--- 条件なしジャンプ
- ---(JMP)--- 条件付きジャンプ
- ---(JMPN)--- ジャンプイフノット

アドレスを示すラベル

ジャンプ命令のアドレスがラベルです。ラベルには最大 4 文字が含まれます。最初の文字はアルファベットで入力し、残りはアルファベットでも数字でも構いません(例：SEG3)。ジャンプラベルは、プログラムの宛先を示します。

宛先を示すラベル

宛先ラベルは、ネットワークの先頭で入力する必要があります。ネットワークの先頭に宛先ラベルを入力するには、ラダーロジックのブラウザから LABEL を選択します。空のボックスが表示されるので、ここにラベル名を入力します。



6.2 ---(JMP)--- 条件なしジャンプ

シンボル

<label name>

---(JMP)

説明

---(JMP)(1 のときにブロック内でジャンプ)は、左側の制御母線と命令の間に他のラダー回路がない場合に、絶対ジャンプを実行します(例を参照)。

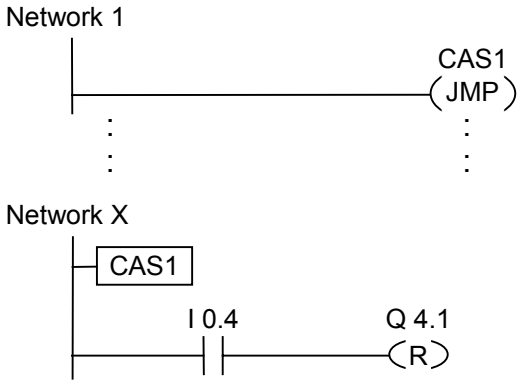
宛先 (LABEL)も---(JMP)ごとに必要となります。

ジャンプ命令とラベル間の命令は実行されません。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	-	-	-	-

例



ジャンプは常に実行され、ジャンプ命令とジャンプラベル間の命令が失われます。

6.3 ---(JMP)--- 条件付きジャンプ

シンボル

<label name>

---(JMP)

説明

---(JMP)(1 のときにブロック内でジャンプ)は、前回の論理演算の RLO が"1"の場合に条件付きジャンプを実行します。

宛先 (LABEL)も---(JMP)ごとに必要となります。

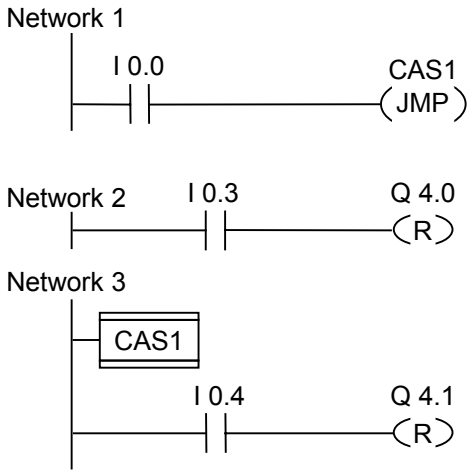
ジャンプ命令とラベル間の命令は実行されません。

条件付きジャンプが実行されない場合、ジャンプ命令の後に RLO は"1"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	1	1	0

例



I0.0 が"1"のとき、ラベル CAS1 へのジャンプが実行されます。ジャンプの性質上、I0.3 のロジック状態が"1"の場合でも、出力 Q4.0 をリセットする命令は実行されません。

6.4 ---(JMPN) ジャンプイフノット

シンボル

<label name>

---(JMPN)

説明

---(JMPN) (ジャンプイフノット)は、RLO が"0"のときに実行される[ラベルへジャンプ]ファンクションに相当します。

宛先 (LABEL)も---(JMPN)ごとに必要となります。

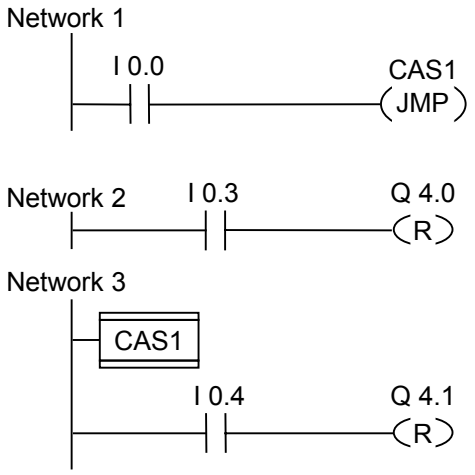
ジャンプ命令とラベル間の命令は実行されません。

条件付きジャンプが実行されない場合、ジャンプ命令の後に RLO は"1"になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	1	1	0

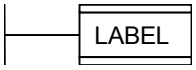
例



I0.0 が"0"のとき、ラベル CAS1 へのジャンプが実行されます。ジャンプの性質上、I0.3 のロジック状態が"1"の場合でも、出力 Q4.0 をリセットする命令は実行されません。

6.5 LABEL ラベル

シンボル



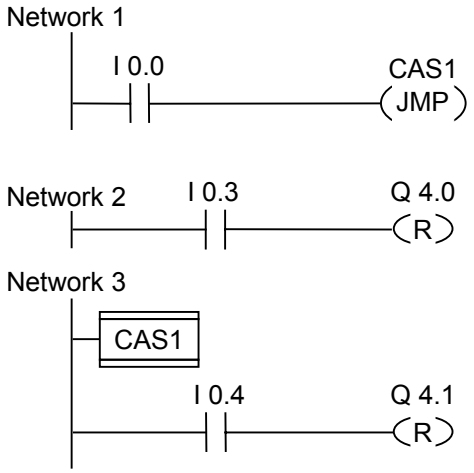
説明

LABEL は、ジャンプ命令の宛先の識別子です。

最初の文字は英字にする必要がありますが、残りは英字でも数字でも構いません(例: CAS1)。

ジャンプラベル(**LABEL**)も指定する必要があります---(JMP)または---(JMPN)。

例



I0.0 が"1"のとき、ラベル CAS1 へのジャンプが実行されます。ジャンプの性質上、I0.3 のロジック状態が"1"の場合でも、出力 Q4.0 をリセットする命令は実行されません。

7 整数演算命令

7.1 整数演算命令の概要

説明

整数演算では、**2 つの整数**(16 ビットおよび 32 ビット)を使用して次の演算を実行することができます。

- ADD_I 整数の加算
- SUB_I 整数の減算
- MUL_I 整数の乗算
- DIV_I 整数の除算

- ADD_DI 倍長整数の加算
- SUB_DI 倍長整数の減算
- MUL_DI 倍長整数の乗算
- DIV_DI 倍長整数の除算
- MOD_DI 倍長整数の商余

7.2 整数演算命令によるステータスワードのビットの評価

説明

整数演算命令は、ステータスワード内の CC1 と CC0、OV と OS の各ビットに影響します。

以下の表に、整数(16 ビットと 32 ビット)を使用した命令の結果に対応するステータスワードのビットの信号状態を示します。

結果の有効範囲	CC 1	CC 0	OV	OS
0	0	0	0	*
16 ビット: -32 768 <= 結果 < 0 (負数) 32 ビット: -2 147 483 648 <=結果 < 0 (負数)	0	1	0	*
16 ビット: 32 767 >= 結果 > 0 (正数) 32 ビット: 2 147 483 647 >= 結果 > 0 (正数)	1	0	0	*

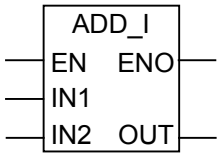
* OS ビットは、この命令結果による影響を受けません。

無効な結果範囲	A1	A0	OV	OS
アンダーフロー(加算) 16 ビット: 結果 = -65536 32 ビット: 結果 = -4 294 967 296	0	0	1	1
アンダーフロー(乗算) 16 ビット: 結果 < -32 768 (負数) 32 ビット: 結果 < -2 147 483 648 (負数)	0	1	1	1
オーバーフロー(加算、減算) 16 ビット: 結果 > 32 767 (正数) 32 ビット: 結果 > 2 147 483 647 (正数)	0	1	1	1
オーバーフロー(乗算、除算) 16 ビット: 結果 > 32 767 (正数) 32 ビット: 結果 > 2 147 483 647 (正数)	1	0	1	1
アンダーフロー(加算、減算) 16 ビット: 結果 < -32 768 (負数) 32 ビット: 結果 < -2 147 483 648 (負数)	1	0	1	1
0 による除算	1	1	1	1

操作	A1	A0	OV	OS
+D: 結果 = -4 294 967 296	0	0	1	1
/D または MOD: 0 による除算	1	1	1	1

7.3 ADD_I 整数の加算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	INT	I、Q、M、L、D または接点	加算の最初の値
IN2	INT	I、Q、M、L、D または接点	加算の 2 番目の値
OUT	INT	I、Q、M、L、D	加算の結果

説明

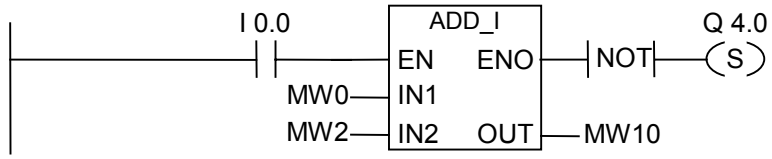
ADD_I (整数の加算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 と IN2 が加算され、その結果は OUT に出力されます。計算結果が、整数(16 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

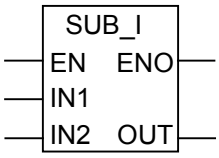
例



ADD_I ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。加算 MW0 + MW2 の結果は、MW10 に出力されます。この結果が整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.4 SUB_I 整数の減算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	INT	I、Q、M、L、D または接点	減算の最初の値
IN2	INT	I、Q、M、L、D または接点	差し引く値
OUT	INT	I、Q、M、L、D	減算の結果

説明

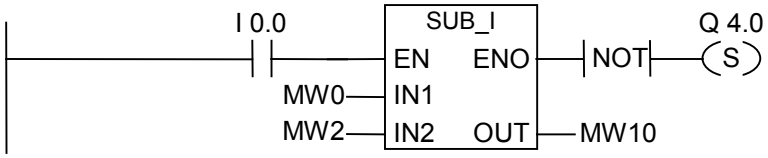
SUB_I (整数の減算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 から IN2 の減算が行われ、その結果は OUT に出力されます。計算結果が、整数(16 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

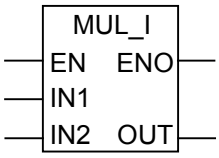
例



SUB_I ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。減算 MW0 - MW2 の結果は、MW10 に出力されます。この結果が整数の許容範囲外の場合か、I0.0 の信号状態が 0 の場合は、出力 Q4.0 が設定されます。

7.5 MUL_I 整数の乗算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	INT	I、Q、M、L、D または接点	乗算の最初の値
IN2	INT	I、Q、M、L、D または接点	乗算の 2 番目の値
OUT	INT	I、Q、M、L、D	乗算の結果

説明

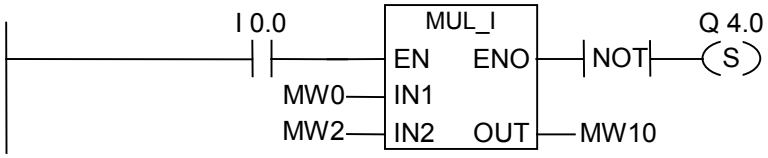
MUL_I (整数の乗算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 と IN2 の乗算が行われ、その結果は OUT に出力されます。計算結果が、整数(16 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

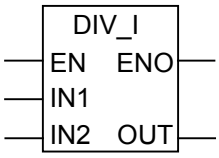
例



MUL_I ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。乗算 MW0 x MW2 の結果は、MW10 に出力されます。この結果が整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.6 DIV_I 整数の除算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	INT	I、Q、M、L、D または接点	被除数
IN2	INT	I、Q、M、L、D または接点	割る値
OUT	INT	I、Q、M、L、D	除算の結果

説明

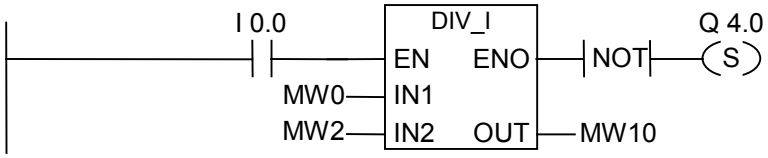
DIV_I (整数の除算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 を IN2 で割り、その結果を OUT に出力します。計算結果が、整数(16 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

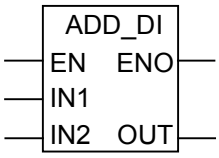
例



DIV_I ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。除算 MW0 ÷ MW2 の結果は、MW10 に出力されます。この結果が整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.7 ADD_DI 倍長整数の加算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DINT	I、Q、M、L、D または接点	加算の最初の値
IN2	DINT	I、Q、M、L、D または接点	加算の 2 番目の値
OUT	DINT	I、Q、M、L、D	加算の結果

説明

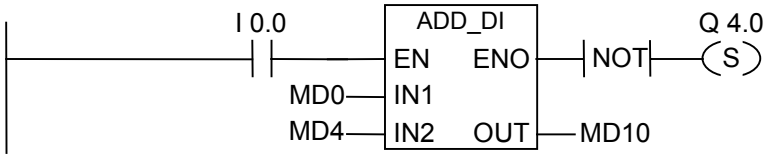
ADD_DI (倍長整数の加算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 と IN2 が加算され、その結果は OUT に出力されます。計算結果が、倍長整数(32 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

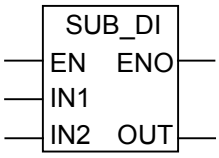
例



ADD_DI ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。加算 MD0 + MD4 の結果は、MD10 に出力されます。この結果が倍長整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.8 SUB_DI 倍長整数の減算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DINT	I、Q、M、L、D または接点	減算の最初の値
IN2	DINT	I、Q、M、L、D または接点	差し引く値
OUT	DINT	I、Q、M、L、D	減算の結果

説明

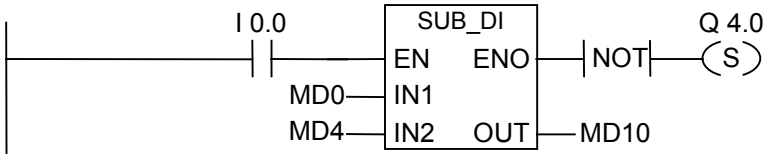
SUB_DI (倍長整数の減算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 から IN2 の減算が行われ、その結果は OUT に出力されます。計算結果が、倍長整数(32 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

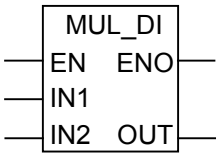
例



SUB_DI ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。減算 MD0 - MD4 の結果は、MD10 に出力されます。この結果が倍長整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.9 MUL_DI 倍長整数の乗算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DINT	I、Q、M、L、D または接点	乗算の最初の値
IN2	DINT	I、Q、M、L、D または接点	乗算の 2 番目の値
OUT	DINT	I、Q、M、L、D	乗算の結果

説明

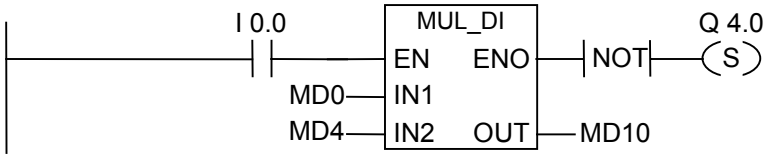
MUL_DI (倍長整数の乗算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 と IN2 の乗算が行われ、その結果は OUT に出力されます。計算結果が、倍長整数(32 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

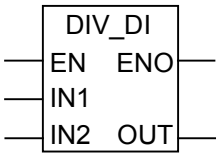
例



MUL_DI ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。乗算 MD0 x MD4 の結果は、MD10 に出力されます。この結果が倍長整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.10 DIV_DI 倍長整数の除算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DINT	I、Q、M、L、D または接点	被除数
IN2	DINT	I、Q、M、L、D または接点	割る値
OUT	DINT	I、Q、M、L、D	除算の結果

説明

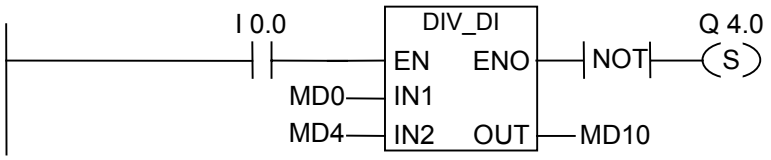
DIV_DI (倍長整数の除算)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 を IN2 で割り、その結果を OUT に出力します。倍長整数の除算エレメントでは、余りは生成されません。計算結果が、倍長整数(32 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

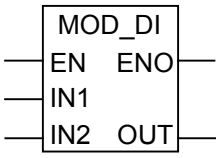
例



DIV_DI ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。除算 MD0 ÷ MD4 の結果は、MD10 に出力されます。この結果が倍長整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

7.11 MOD_DI 倍長整数の商余

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DINT	I、Q、M、L、D または接点	被除数
IN2	DINT	I、Q、M、L、D または接点	割る値
OUT	DINT	I、Q、M、L、D	除算の余り

説明

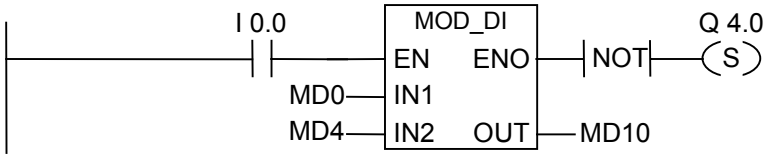
MOD_DI (倍長整数の商余)は、イネーブル(EN)入力のロジック状態が"1"のときに起動します。IN1 を IN2 で割り、その結果を OUT に出力します。計算結果が、倍長整数(32 ビット)の有効範囲にない場合、OV ビットと OS ビットは"1"になり、ENO は"0"になるため、この計算ボックス以降の ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: 整数演算命令によるステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

例



MOD_DI ボックスは、I0.0 の信号状態が"1"になると、アクティブ化されます。除算 MD0÷MD4 の剰余は、MD10 に出力されます。この商余が倍長整数の許容範囲外の場合は、出力 Q4.0 が設定されます。

8 浮動小数点数値演算命令

8.1 浮動小数点数値演算命令の概要

説明

IEEE 32 ビット浮動小数点数は、REAL と呼ばれるデータタイプに属します。浮動小数点数値演算命令を使用し、**32 ビット IEEE 浮動小数点数を 2 つ使用すれば**、以下の数値演算命令を実行できます。

- ADD_R　実数の加算
- SUB_R　実数の減算
- MUL_R　実数の乗算
- DIV_R　実数の除算

浮動小数点数値演算を使用し、**32 ビット IEEE 浮動小数点数**を **1 つ**使用すれば、以下の操作を実行できます。

- 絶対値(ABS)を求める
- 平方(SQR)および平方根(SQRT)を求める
- 自然対数(LN)を求める
- 基数 e(= 2,71828)に対する指数値(EXP)を求める
- 32 ビット IEEE 浮動小数点数で表す角について、次の三角関数を求める
  - サイン(SIN)およびアークサイン(ASIN)
  - コサイン(COS)およびアークコサイン(ACOS)
  - タンジェント(TAN)およびアークタンジェント(ATAN)

関連項目: ステータスワードのビットの評価

8.2 浮動小数点数値演算命令におけるステータスワードのビットの評価

説明

浮動小数点命令は、ステータスワード内の CC 1 と CC 0、OV と OS の各ビットに影響します。

以下の表に、浮動小数点数(32 ビット)を使用した命令の結果に対応するステータスワードのビットの信号状態を示します。

有効な結果範囲	CC 1	CC 0	OV	OS
＋0, －0(ゼロ)	0	0	0	*
-3.402823E+38 < 結果 < -1.175494E-38 (負数)	0	1	0	*
+1.175494E-38 < 結果 < 3.402824E+38 (正数)	1	0	0	*

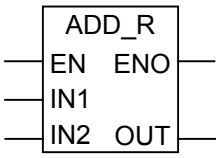
* OS ビットは、この命令結果による影響を受けません。

結果の有効範囲	CC 1	CC 0	OV	OS
アンダーフロー -1.175494E-38 < 結果 < - 1.401298E-45 (負数)	0	0	1	1
アンダーフロー +1.401298E-45 < 結果< +1.175494E-38 (正数)	0	0	1	1
オーバーフロー 結果< -3.402823E+38 (負の数)	0	1	1	1
オーバーフロー 結果> 3.402823E+38 (正数)	1	0	1	1
無効な実数または不正な命令 (有効範囲にない入力値)	1	1	1	1

8.3 基本命令

8.3.1 ADD_R 実数の加算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	REAL	I、Q、M、L、D または接点	加算の最初の値
IN2	REAL	I、Q、M、L、D または接点	加算の 2 番目の値
OUT	REAL	I、Q、M、L、D	加算の結果

説明

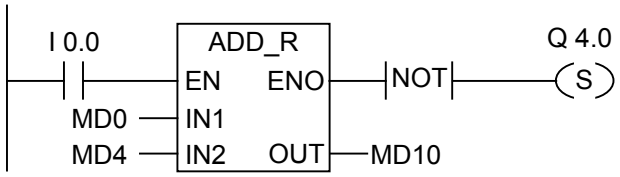
ADD_R(実数の加算)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。IN1 と IN2 が加算され、その結果は OUT に出力されます。演算結果が実数の有効範囲にない場合(オーバーフローまたはアンダーフロー)、OV ビットと OS ビットは "1"になり、ENO は"0"になるため、この演算ボックス以降の、ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

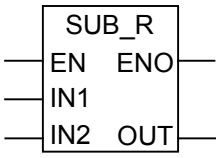
例



ADD_R ボックスは、I0.0 のロジック"1"によってアクティブ化されます。加算 MW0+MW4 の結果は、MW10 に出力されます。この結果が浮動小数点数の許容範囲外の場合か、プログラムステートメントが処理されなかった場合は(I0.0 = 0)、出力 Q4.0 が設定されます。

8.3.2 SUB_R 実数の減算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	REAL	I、Q、M、L、D または接点	減算の最初の値
IN2	REAL	I、Q、M、L、D または接点	差し引く値
OUT	REAL	I、Q、M、L、D	減算の結果

説明

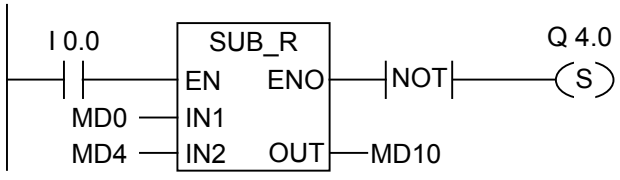
SUB_R(実数の減算)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。IN1 から IN2 の減算が行われ、その結果は OUT に出力されます。演算結果が実数の有効範囲にない場合(オーバーフローまたはアンダーフロー)、OV ビットと OS ビットは "1"になり、ENO は"0"になるため、この演算ボックス以降の、ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

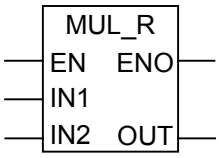
例



SUB_R ボックスは、I0.0 のロジック"1"によってアクティブ化されます。減算 MD0 - MD4 の結果は、MD10 に出力されます。この結果が浮動小数点数の許容範囲外の場合か、プログラムステートメントが処理されなかった場合は、出力 Q4.0 が設定されます。

8.3.3 MUL_R 実数の乗算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	REAL	I、Q、M、L、D または接点	乗算の最初の値
IN2	REAL	I、Q、M、L、D または接点	乗算の 2 番目の値
OUT	REAL	I、Q、M、L、D	乗算の結果

説明

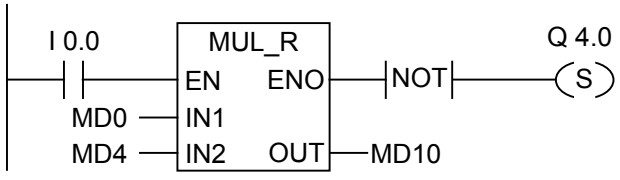
MUL_R(実数の乗算)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。IN1 と IN2 の乗算が行われ、その結果は OUT に出力されます。演算結果が実数の有効範囲にない場合(オーバーフローまたはアンダーフロー)、OV ビットと OS ビットは "1"になり、ENO は"0"になるため、この演算ボックス以降の、ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

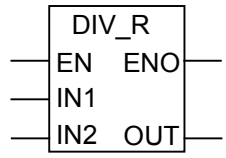
例



MUL_R ボックスは、I0.0 のロジック"1"によってアクティブ化されます。乗算 MD0 x MD4 の結果は、MD0 に出力されます。この結果が浮動小数点数の許容範囲外の場合か、プログラムステートメントが処理されなかった場合は、出力 Q4.0 が設定されます。

8.3.4 DIV_R 実数の除算

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	REAL	I、Q、M、L、D または接点	被除数
IN2	REAL	I、Q、M、L、D または接点	割る値
OUT	REAL	I、Q、M、L、D	除算の結果

説明

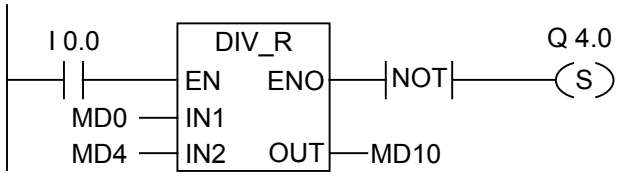
DIV_R(実数の除算)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。IN1 を IN2 で割り、その結果を OUT に出力します。演算結果が実数の有効範囲にない場合(オーバーフローまたはアンダーフロー)、OV ビットと OS ビットは "1"になり、ENO は"0"になるため、この演算ボックス以降の、ENO に接続されている他のファンクション(カスケード配列)は、実行されません。

関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

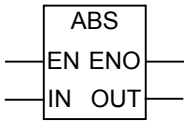
例



DIV_R ボックスは、I0.0 のロジック"1"によってアクティブ化されます。除算 MD0÷MD4 の結果は、MD10 に出力されます。この結果が浮動小数点数の許容範囲外の場合か、プログラムステートメントが処理されなかった場合は、出力 Q4.0 が設定されます。

8.3.5 ABS 浮動小数点数の絶対値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数の絶対値

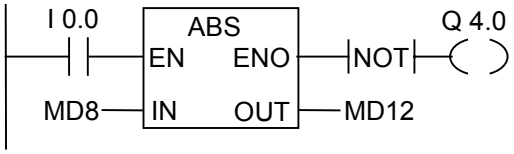
説明

ABS は、浮動小数点数の絶対値を求めます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

例



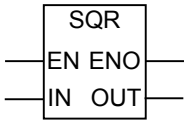
I0.0 が"1"のとき、MD8 の絶対値が MD12 に出力されます。

MD8 = + 6.234 のとき、MD12 = 6.234 になります。変換が実行されないと、出力 Q4.0 は"1"になります(ENO = EN = 0)。

8.4 拡張命令

8.4.1 SQR 平方を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数の平方

説明

SQR は、浮動小数点数の平方を演算します。

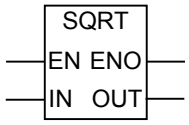
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.2 SQRT 平方根を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数の平方根

説明

SQRT は、浮動小数点数の平方根を演算します。この命令は、アドレスが"0"より大きい場合に正の結果になります。唯一の例外: -0 の平方根は-0 です。

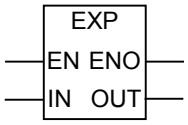
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.3 EXP 指数値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数の指数値

説明

EXP は、e(=2,71828...)を基本として、浮動小数点数の指数値を演算します。

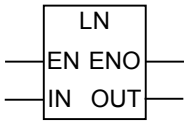
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.4 LN 自然対数を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数の自然対数

説明

LN は、浮動小数点数の自然対数を演算します。

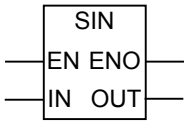
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.5 SIN サイン値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のサイン

説明

SIN は、浮動小数点数のサイン値を演算します。浮動小数点数は、ここではラジアン測定値の角度を示します。

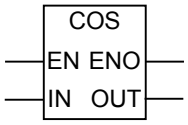
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.6 COS コサイン値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のコサイン

説明

COS は、浮動小数点数のコサインを演算します。浮動小数点数は、ここではラジアン測定値の角度を示します。

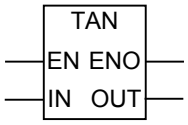
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.7 TAN タンジェント値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のタンジェント

説明

TAN は、浮動小数点数のタンジェントを演算します。浮動小数点数は、ここではラジアン測定値の角度を示します。

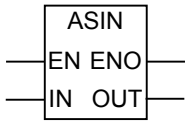
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.8 ASIN アークサイン値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のアークサイン

説明

ASIN は、定義範囲が-1 <= 入力値 <= 1 の浮動小数点数のアークサインを演算します。この結果は、範囲内のラジアン単位の角度で示されます。

$$-\pi/2 \leq \text{output value} \leq +\pi/2$$

where π = 3.1415....

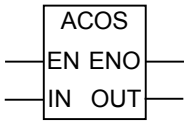
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.9 ACOS アークコサイン値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のアークコサイン

説明

ACOS は、定義範囲が-1 <= 入力値 <= 1 の浮動小数点数のアークサインを演算します。この結果は、範囲内のラジアン単位の角度で示されます。

$$0 \leq \text{output value} \leq +\pi$$

where π = 3.1415....

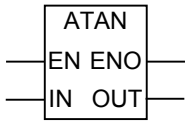
関連項目: ステータスワードのビットの評価

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

8.4.10 ATAN アークタンジェント値を求める

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	REAL	I、Q、M、L、D または接点	入力値: 浮動小数点
OUT	REAL	I、Q、M、L、D	出力値: 浮動小数点数のアークタンジェント

説明

ATAN は、浮動小数点数のアークタンジェント値です。これにより、範囲内のラジアン測定値の角度が得られます。

$$-\pi/2 \leq \text{output value} \leq +\pi/2$$

where π = 3.1415....

関連項目: ステータスワードのビットの評価

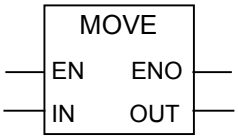
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	x	0	x	x	1

9 移動命令

9.1 MOVE 値の割り付け

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	8 ビット長、16 ビット長、または 32 ビット長のすべての基本データタイプ	I、Q、M、L、D または接点	元の値
OUT	8 ビット長、16 ビット長、または 32 ビット長のすべての基本データタイプ	I、Q、M、L、D	宛先アドレス

説明

MOVE (値の割り付け)は、イネーブル EN 入力で起動します。IN 入力で指定した値が、出力 OUT で指定したアドレスにコピーされます。ENO は、EN と同じロジック状態になります。**MOVE** は、データオブジェクト BYTE、WORD、または DWORD のみをコピーできます。配列やストラクチャなどのユーザー定義データタイプは、システムファンクション"BLKMOVE"(SFC 20)を使用してコピーする必要があります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	-	-	-	-	0	1	1	1

MCR (マスタコントロールリレー)の依存性

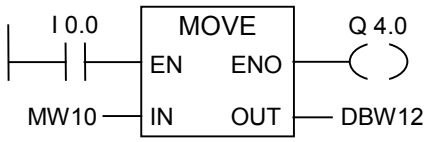
MCR の依存性は、Move ボックスがアクティブな MCR ゾーン内に配置された場合にのみ有効になります。アクティブな MCR ゾーンで、アドレス指定されているデータが上記の方法でコピーされるのは、MCR がオンになっており、かつ、イネーブル入力へ電力の流れがある場合です。MCR がオフのときに MOVE が実行されると、現在の IN の状態に関係なく、指定された OUT アドレスにロジック"0"が書き込まれます。

注記

長さの異なるデータタイプに値を移動すると、必要に応じて高位値のバイトが切り捨てられるか、ゼロが充てんされます。

例: ダブルワード	**1111 1111**	**0000 1111**	**1111 0000**	**0101 0101**
移動	**結果**			
ダブルワードへの変換	1111 1111	0000 1111	1111 0000	0101 0101
バイトに移動				0101 0101
ワードへの変換			1111 0000	0101 0101
例: バイト				**1111 0000**
移動	**結果**			
バイトに移動				1111 0000
ワードへの変換			0000 0000	1111 0000
ダブルワードへの変換	0000 0000	0000 0000	0000 0000	1111 0000

例



入力 I0.0 が 1 になると、命令が実行され、MW10 の内容が現在のオープン DB のデータワード 12(DBW12)にコピーされます。

命令が実行されると、Q4.0 は"1"になります。

例に示す回路がアクティブな MCR ゾーン内にある場合、以下が実行されます。

- MCR がオンのとき、前述のように、MW10 のデータが DBW12 にコピーされます。
- MCR がオフのとき、DBW12 に"0"が書き込まれます。

10 プログラム制御命令

10.1 プログラム制御命令の概要

説明

使用可能なプログラム制御命令を次に示します。

- ---(CALL)　FC SFC のコイルからの呼び出し(パラメータなし)
- CALL_FB　FB のボックスからの呼び出し
- CALL_FC　FC のボックスからの呼び出し
- CALL_SFB　SFB のボックスからの呼び出し
- CALL_SFC　SFC のボックスからの呼び出し
- 複数インスタンスの呼び出し
- ライブラリからのブロックの呼び出し

- MCR ファンクションの使用方法に関する重要事項
- ---(MCR<)　マスタコントロールリレーのオン
- ---(MCR>)　マスタコントロールリレーのオフ
- ---(MCRA)　マスタコントロールリレーの開始
- ---(MCRD)　マスタコントロールリレーの終了

- RET　リターン

10.2 ---(Call) FC SFC のコイルからの呼び出し(パラメータなし)

シンボル

<FC/SFC 番号>

---(CALL)

パラメータ	データタイプ	メモリ領域	説明
<FC/SFC 番号>	BLOCK_FC BLOCK_SFC	-	FC/SFC の番号。範囲は CPU により異なる

説明

---(Call) (パラメータなしの、FC または SFC 呼び出し)は、パラメータが指定されていないファンクション(FC)またはシステムファンクション(SFC)を呼び出す場合に使用されます。CALL コイルの RLO が"1"の場合に呼び出しが実行されます。**---(Call)** が実行されると、次の処理が実行されます。

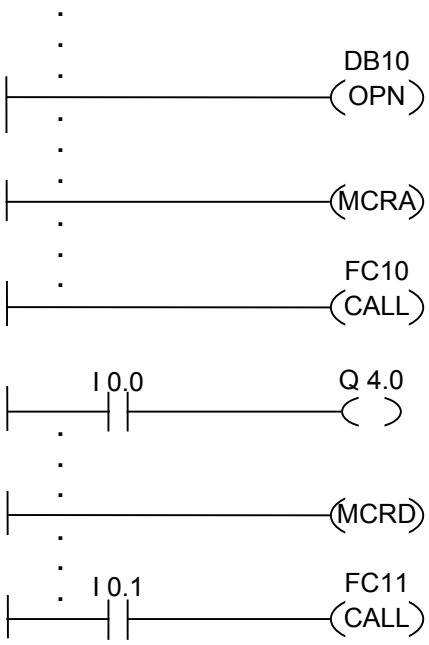
- 呼び出し側ブロックのリターンアドレスが保存されます。
- 前のローカルデータ領域が、現在のローカルデータ領域に置き換えられます。
- MA ビット(有効な MCR ビット)は、B スタックにシフトされます。
- 呼び出されたファンクションに対し、新しいローカルデータ領域が作成されます。

このあと、呼び出された FC または SFC でプログラム処理が実行されます。

ステータスワード

		BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
条件なし	書き込みの内容:	-	-	-	-	0	0	1	-	0
条件付き:	書き込みの内容:	-	-	-	-	0	0	1	1	0

例



上記のラダー回路は、ユーザーが作成したファンクションブロックのプログラムセクションです。この FB では、DB10 が開き、MCR の機能が有効になります。FC10 の条件なしの呼び出しが実行されると、次の状態が発生します。

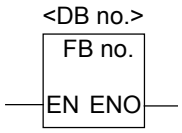
呼び出し側 FB のリターンアドレス、DB10 の選択データ、呼び出し側 FB のインスタンスデータブロックの選択データが保存されます。MCRA 命令で"1"に設定された MA ビットは、B スタックにプッシュされ、呼び出されたブロック(FC10)に対し"0"を設定します。プログラム処理は FC10 で続行されます。FC10 で MCR の機能が必要な場合は、FC10 内で MCR を再度有効化する必要があります。FC10 が終了すると、プログラム処理は呼び出し側 FB に戻ります。どの DB FC10 が使用されているかに関係なく、MA ビットがリストアされ、DB10 と、ユーザーが作成した FB のインスタンスデータブロックが現在の DB に戻ります。プログラムは続行して次の回路に進み、I0.0 の論理状態を出力 Q4.0 に割り付けます。FC11 の呼び出しは条件付き呼び出しです。これは、I0.1 が"1"の場合にのみ実行されます。実行されると、FC11 とのプログラム制御のやりとりについて、FC10 で説明したプロセスと同じプロセスが実行されます。

注記

呼び出し側のブロックに戻った後に、前に開いていた DB が再び開くとは限りません。詳細については、Readme ファイルをお読みください。

10.3 CALL_FB FB のボックスからの呼び出し

シンボル



シンボルは、FB によって異なります(パラメータの指定の有無や FC の数によって異なる)。EN、ENO、および FB の名前または番号を指定する必要があります。

パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
FB 番号 DB 番号	BLOCK_FB BLOCK_DB	- -	FB/DB の番号。範囲は CPU により異なる

説明

CALL_FB (ファンクションブロックのボックスからの呼び出し)は、EN が"1"になると実行されます。CALL_FB が実行されると、

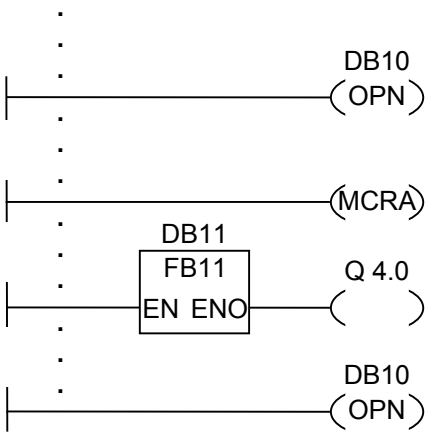
- 呼び出し側ブロックのリターンアドレスが保存されます。
- 2 つの現在のデータブロック(DB およびインスタンス DB)の選択データが保存されます。
- 前のローカルデータ領域が、現在のローカルデータ領域に置き換えられます。
- MA ビット(有効な MCR ビット)は、B スタックにシフトされます。
- 呼び出されたファンクションブロックに対し、新しいローカルデータ領域が作成されます。

このあと、呼び出されたファンクションブロックでプログラム処理が継続されます。ENO を検出するために、BR ビットを確認します。ユーザーは、---(SAVE)を使って、呼び出されたブロックの BR ビットに、必要な状態(エラー評価)を割り付ける必要があります。

ステータスワード

		BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
条件なし	書き込みの内容:	x	-	-	-	0	0	x	x	x
条件付き:	書き込みの内容:	-	-	-	-	0	0	x	x	x

例



上記のラダー回路は、ユーザーが作成したファンクションブロックのプログラムセクションです。この FB では、DB10 が開き、MCR の機能が有効になります。条件なしで FB11 の呼び出しが実行されると、次の状態が発生します。

呼び出し側 FB のリターンアドレス、DB10 の選択データ、呼び出し側 FB のインスタンスデータブロックの選択データが保存されます。MCRA 命令で"1"に設定された MA ビットは、B スタックにプッシュされ、呼び出されたブロック(FB11)に対し"0"を設定します。プログラム処理は FB11 で続行されます。FB11 で MCR の機能が必要な場合は、FB11 内で MCR を再度有効化する必要があります。呼び出し元 FB でエラーを評価できるように、---(SAVE)命令を使用して、RLO の状態を BR ビットに保存する必要があります。FB11 が終了すると、プログラム処理は呼び出し側 FB に戻ります。MA ビットはリストアされ、ユーザーが作成した FB のインスタンスデータブロックが再び開きます。FB11 が正常に処理されると、ENO = "1"になり、したがって Q4.0 = "1"になります。

注記

FB または SFB を開くと、前に開いていた DB の番号は失われます。必要な DB は、再度開く必要があります。

10.4 CALL_FC FC のボックスからの呼び出し

シンボル

シンボルは、FC によって異なります(パラメータ指定の有無や FC の数により異なる)。EN、ENO、および FC の名前または番号を指定する必要があります。

パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
FC 番号	BLOCK_FC	-	FC の番号。範囲は CPU により異なる

説明

CALL_FC (ファンクションのボックスからの呼び出し)は、ファンクション(FC)を呼び出す場合に使用します。呼び出しは、EN が"1"になると実行されます。CALL_FC が実行されると、

- 呼び出し側ブロックのリターンアドレスが保存されます。
- 前のローカルデータ領域が、現在のローカルデータ領域に置き換えられます。
- MA ビット(有効な MCR ビット)は、B スタックにシフトされます。
- 呼び出されたファンクションに対し、新しいローカルデータ領域が作成されます。

このあと、呼び出されたファンクションでプログラム処理が継続されます。

ENO を検出するために、BR ビットを確認します。ユーザーは、**---(SAVE)**を使って、呼び出されたブロックの BR ビットに、必要な状態(エラー評価)を割り付ける必要があります。

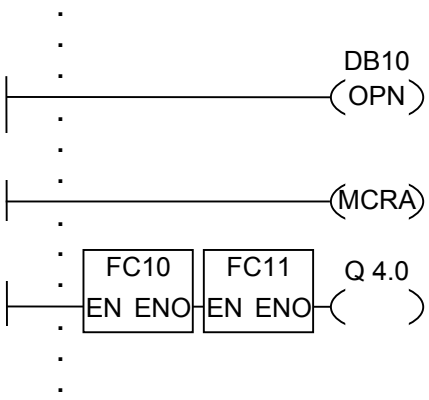
ファンクションの呼び出しを行い、呼び出されたブロックの変数宣言テーブルに IN、OUT、および IN_OUT の宣言がある場合、これらの変数は、呼び出し側ブロックのプログラムに仮パラメータリストとして追加されます。

ファンクションを呼び出す場合、その呼び出し位置で実パラメータを仮パラメータに割り付ける**必要**があります。どのような場合でも、ファンクション宣言内に初期値を指定しても、意味がありません。

ステータスワード

		BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
条件なし	書き込みの内容:	x	-	-	-	0	0	x	x	x
条件付き:	書き込みの内容:	-	-	-	-	0	0	x	x	x

例



上記のラダー回路は、ユーザーが作成したファンクションブロックのプログラムセクションです。この FB では、DB10 が開き、MCR の機能が有効になります。FC10 の条件なしの呼び出しが実行されると、次の状態が発生します。

呼び出し側 FB のリターンアドレス、DB10 の選択データ、呼び出し側 FB のインスタンスデータブロックの選択データが保存されます。MCRA 命令で"1"に設定された MA ビットは、B スタックにプッシュされ、呼び出されたブロック(FC10)に対し"0"を設定します。プログラム処理は FC10 で続行されます。FC10 で MCR の機能が必要な場合は、FC10 内で MCR を再度有効化する必要があります。呼び出し元 FB でエラーを評価できるように、---(SAVE)命令を使用して、RLO の状態を BR ビットに保存する必要があります。FC10 が終了すると、プログラム処理は呼び出し側 FB に戻ります。MA ビットが復元されます。FC10 が実行されると、ENO に応じて、呼び出し側 FB でプログラム処理が継続されます。

ENO = "1" FC11 の処理が行われます。

ENO = "0" 次のネットワークで処理が開始します。

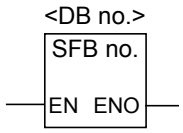
FC11 が正しく処理された場合も、ENO = "1"になり、したがって Q**4.0 = "1"**になります。

注記

呼び出し側のブロックに戻った後に、前に開いていた DB が再び開くとは限りません。詳細については、Readme ファイルをお読みください。

10.5 CALL_SFB SFB のボックスからの呼び出し

シンボル



シンボルは、SFB によって異なります(パラメータ指定の有無や FC の数により異なる)。EN、ENO、および SFB の名前または番号を指定する必要があります。

パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
SFB 番号 DB 番号	ブロック_SFB BLOCK_DB	- -	SFB の番号。範囲は CPU により異なる

説明

CALL_SFB (システムファンクションブロックのボックスからの呼び出し)は、EN が"1"になると実行されます。CALL_SFB が実行されると、

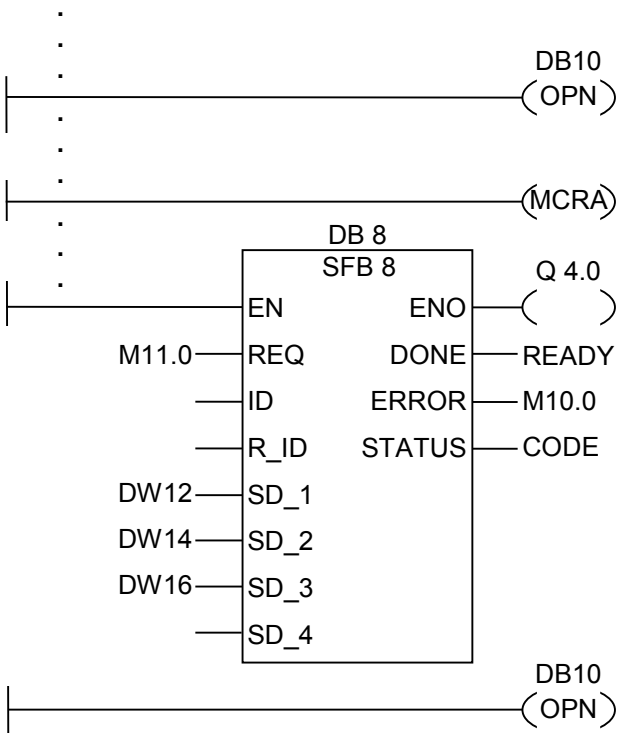
- 呼び出し側ブロックのリターンアドレスが保存されます。
- 2 つの現在のデータブロック(DB およびインスタンス DB)の選択データが保存されます。
- 前のローカルデータ領域が、現在のローカルデータ領域に置き換えられます。
- MA ビット(有効な MCR ビット)は、B スタックにシフトされます。
- 呼び出されたシステムファンクションブロック対し、新しいローカルデータ領域が作成されます。

呼び出された SFB でプログラム処理が継続されます。SFB が呼び出されると(EN = "1")、ENO は"1"になります。エラーは発生しません。

ステータスワード

		BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
条件なし	書き込みの内容:	x	-	-	-	0	0	x	x	x
条件付き:	書き込みの内容:	-	-	-	-	0	0	x	x	x

例



上記のラダー回路は、ユーザーが作成したファンクションブロックのプログラムセクションです。この FB では、DB10 が開き、MCR の機能が有効になります。SFB8 の条件なしの呼び出しが実行されると、次の状態が発生します。

呼び出し側 FB のリターンアドレス、DB10 の選択データ、呼び出し側 FB のインスタンスデータブロックの選択データが保存されます。MCRA 命令で"1"に設定された MA ビットは、B スタックにプッシュされ、呼び出されたブロック(SFB8)に対し"0"を設定します。プログラム処理は SFB8 で続行されます。SFB8 が終了すると、プログラム処理は呼び出し側 FB に戻ります。MA ビットがリストアされ、ユーザーが作成した FB のインスタンスデータブロックが、現在のインスタンス DB になります。SFB8 が正常に処理されると、ENO = "1"になり、したがって Q4.0 = "1"になります。

注記

FB または SFB を開くと、前に開いていた DB の番号は失われます。必要な DB は、再度開く必要があります。

10.6 CALL_SFC SFC のボックスからの呼び出し

シンボル

シンボルは、SFC によって異なります(パラメータ指定の有無や FC の数により異なる)。EN、ENO、および SFC の名前または番号を指定する必要があります。

パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	-	イネーブル入力
ENO	BOOL	-	イネーブル出力
SFC 番号	BLOCK_SFC	-	SFC の番号。範囲は CPU により異なる

説明

CALL_SFC (システムファンクションのボックスからの呼び出し)は、SFC を呼び出す場合に使用されます。呼び出しは、EN が"1"になると実行されます。CALL_SFC が実行されると、

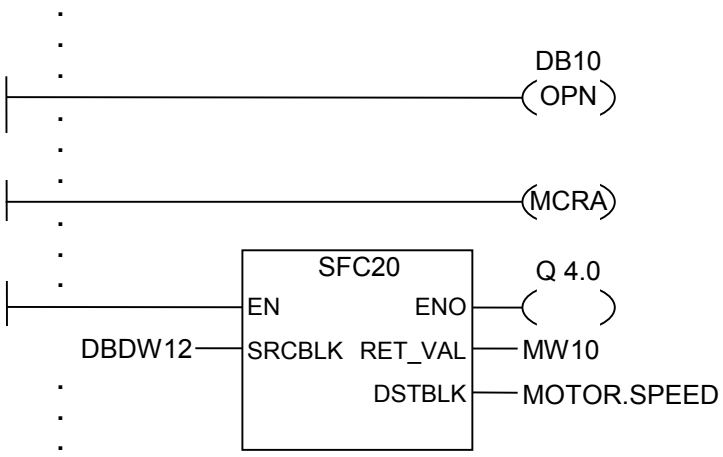
- 呼び出し側ブロックのリターンアドレスが保存されます。
- 前のローカルデータ領域が、現在のローカルデータ領域に置き換えられます。
- MA ビット(有効な MCR ビット)は、B スタックにシフトされます。
- 呼び出されたシステムファンクションに対し、新しいローカルデータ領域が作成されます。

このあと、呼び出された SFC でプログラム処理が実行されます。SFC が呼び出されると(EN = "1")、ENO は"1"になります。エラーは発生しません。

ステータスワード

		BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
条件なし	書き込みの内容:	x	-	-	-	0	0	x	x	x
条件付き:	書き込みの内容:	-	-	-	-	0	0	x	x	x

例



上記のラダー回路は、ユーザーが作成したファンクションブロックのプログラムセクションです。この FB では、DB10 が開き、MCR の機能が有効になります。SFC20 の条件なしの呼び出しが実行されると、次の状態が発生します。

呼び出し側 FB のリターンアドレス、DB10 の選択データ、呼び出し側 FB のインスタンスデータブロックの選択データが保存されます。MCRA 命令で"1"に設定された MA ビットは、B スタックにプッシュされ、呼び出されたブロック(SFC20)に対し"0"を設定します。プログラム処理は SFC20 で続行されます。SFC20 が終了すると、プログラム処理は呼び出し側 FB に戻ります。MA ビットが復元されます。

SFC20 が処理された後、ENO に応じて、呼び出し側 FB でプログラムが引き続き実行されます。

ENO = "1" Q4.0 = "1"

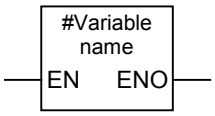
ENO = "0" Q4.0 = "0"

注記

呼び出し側のブロックに戻った後に、前に開いていた DB が再び開くとは限りません。詳細については、Readme ファイルをお読みください。

10.7 複数インスタンスの呼び出し

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
#Variable name	FB、SFB	-	複数インスタンスの名前

説明

ファンクションブロックのデータタイプを使ってスタティック変数を宣言することで、複数のインスタンスが作成されます。すでに宣言されている複数インスタンスのみが、プログラム要素のカタログに組み込まれます。複数インスタンスのシンボルは、パラメータが指定されているかどうか、指定されている場合はその数に応じて異なります。EN、ENO、および変数名は常に指定されています。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	0	0	x	x	x

10.8 ライブラリからのブロックの呼び出し

SIMATIC Manager で使用可能なライブラリを使用すれば、次のブロックを選択できます。

- CPU オペレーティングシステムに統合されているブロック(バージョン 3 の STEP 7 プロジェクトの場合は"標準ライブラリ"、バージョン 2 の STEP 7 プロジェクトの場合は"stdlibs(V2)")
- 繰り返し使用できるように、あらかじめライブラリに保存してあるブロック

10.9 MCR ファンクションの使用方法に関する重要事項

マスタコントロールリレーを MCRA で起動したときに、その起動元となったブロックには注意が必要です。

- MCR が無効になると、---(MCR<) と---(MCR>)間のプログラムセグメント内の全割り付けにより値 0 が書き込まれます。これは、ブロックへのパラメータ転送などの割り付けが入った**すべての**ボックスに対して有効です。
- **MCR** 命令の実行前に RLO が 0 だった場合、MCR は無効になります。

危険:PLC が STOP になったり、未定義のランタイム特性が生成されます!

コンパイラは、VAR_TEMP で定義されているテンポラリ変数のローカルデータへ書き込みアクセスして、アドレスの計算を行います。すなわち、次のコマンドシーケンスを使用すると、PLC が STOP に設定されたり、ランタイム特性が未定義になったりします。

仮パラメータアクセス

- タイプ STRUCT、UDT、ARRAY、STRING の複合 FC パラメータのコンポーネントにアクセス
- マルチプルインスタンス能力を持つブロック(バージョン 2 ブロック)の IN_OUT 領域からの、STRUCT、UDT、ARRAY、STRING タイプの複合 FB パラメータで構成されるコンポーネントへのアクセス
- アドレスが 8180.0.を超える場合の、マルチプルインスタンス能力を持つファンクションブロック(バージョン 2 ブロック)のパラメータへのアクセス
- マルチプルインスタンス能力を持つファンクションブロック(バージョン 2 ブロック)の DB0 を開く BLOCK_DB パラメータへのアクセス。さらにデータアクセスが実行されると、CPU は STOP になります。T 0、C 0、FC0、または FB0 も、必ず TIMER、COUNTER、BLOCK_FC、および BLOCK_FB に使用されます。

パラメータの引き渡し

- パラメータが転送される呼び出し

LAD/FBD

- RLO = 0 で開始する Ladder または FBD の T ブランチおよび中間出力

対策

上記のコマンドを MCR から独立させます。

1st 該当するステートメントまたはネットワークの前で、**マスタコントロールリレーの終了**命令を使用して、マスタコントロールリレーを無効にします。

2nd 該当するステートメントまたはネットワークの前に**マスタコントロールリレーの開始**命令を使用して、マスタコントロールリレーを有効にします。

10.10 ---(MCR<) マスタコントロールリレーのオン

MCR ファンクションの使用方法に関する重要事項

シンボル

---(MCR<)

説明

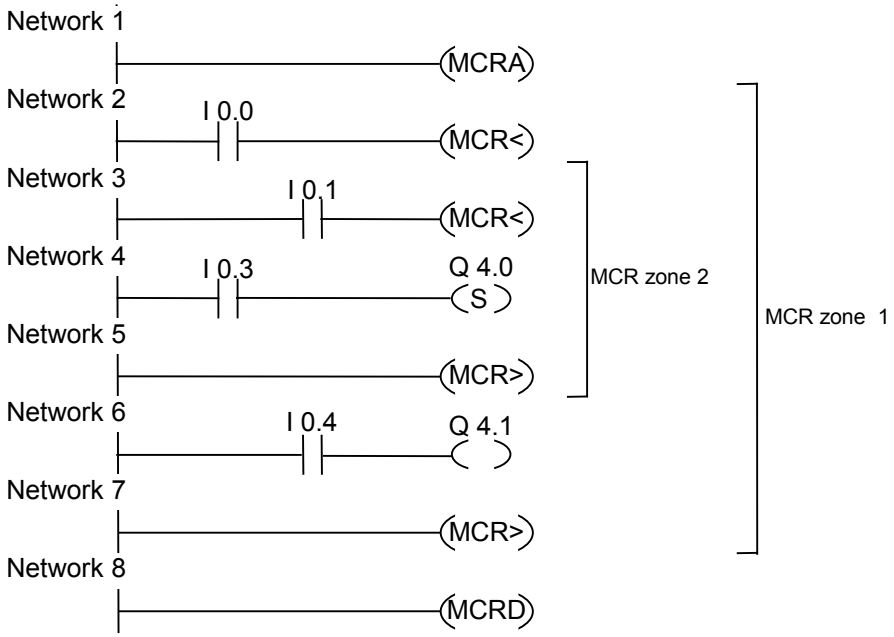
---(MCR<)(マスタコントロールリレーのオン)は、RLO を MCR スタックに保存します。MCR ネスティングスタックは、LIFO(後入れ先出し)方式のスタックで、8 段階のスタック(ネスティングレベル)が可能です。スタックがいっぱいの場合、---(MCR<)ファンクションは MCR スタックエラー(MCRF)を生成します。次に示す要素は、MCR に依存し、RLO 状態の影響を受けます。RLO の状態は、MCR ゾーンが開いているときに MCR スタックに保存されます。

- --(#) ミッドライン出力
- --( ) 出力
- --(S) セット出力
- --(R) リセット出力
- RS リセットフリップフロップ
- SR セットフリップフロップ
- MOVE 値の割り付け

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	1	-	0

例



MCR の機能は、MCRA 回路によって有効になります。これにより、最大で 8 段階にネストされた MCR ゾーンを作成できます。この例では、2 つの MCR ゾーンがあり、次のようにファンクションが実行されます。

I0.0 = "1" (MCR はゾーン 1 のオン): I0.4 の論理状態が Q4.1 に割り付けられます。

I0.0 = "0" (MCR はゾーン 1 のオフ): I0.4 の論理状態に関わらず、Q4.1 は"0"になります。

I0.1 = "1" (MCR はゾーン 2 のオン): I0.3 が"1"の場合は、Q4.0 が"1"に設定されます。

I0.1 = "0" (MCR はゾーン 2 のオフ): I0.3 の論理状態に関わらず、Q4.0 は変わりません。

10.11 ---(MCR>) マスタコントロールリレーのオフ

MCR ファンクションの使用方法に関する重要事項

シンボル

---(MCR>)

説明

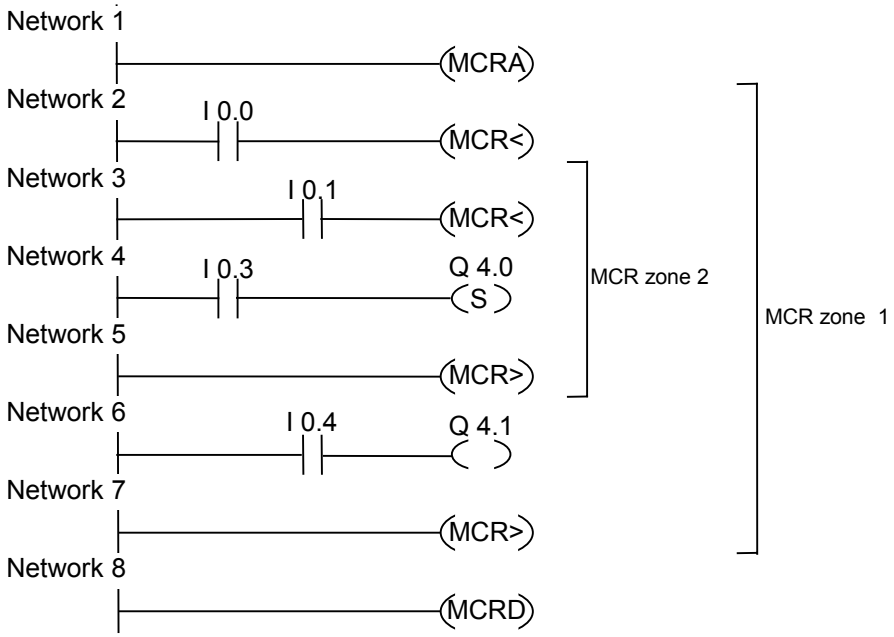
---(MCR>)(開いているMCRゾーンをクローズ)は、MCRスタックからRLOエントリを削除します。MCR ネスティングスタックは、LIFO(後入れ先出し)方式のスタックで、8 段階のスタック(ネスティングレベル)が可能です。スタックが空の場合は、MCR スタックエラー(MCRF)が生成されます。次の要素は、MCR に依存し、RLO 状態の影響を受けます。RLO の状態は、MCR ゾーンが開いているときに MCR スタックに保存されます。

- --(#) ミッドライン出力
- --( ) 出力
- --(S) セット出力
- --(R) リセット出力
- RS リセットフリップフロップ
- SR セットフリップフロップ
- MOVE 値の割り付け

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	1	-	0

例



MCR の機能は、---(MCRA)ラダー回路により有効になります。これにより、最大で 8 段階にネストされた MCR ゾーンを作成できます。この例では、2 つの MCR ゾーンがあり、最初の---(MCR>) (MCR OFF)ラダー回路は、2 番目の---(MCR<) (MCR ON)ラダー回路に属しています。この 2 つのゾーン間のラダー回路はすべて、MCR ゾーン 2 に属します。ファンクションは、次のように実行されます。

I0.0 = "1": I0.4 の論理状態が Q4.1 に割り付けられます。

I0.0 = "0": I0.4 の論理状態に関わらず、Q4.1 は"0"になります。

I0.1 = "1": I0.3 が"1"の場合は、Q4.0 が"1"に設定されます。

I0.1 = "0": I0.3 の論理状態に関わらず、Q4.0 は変わりません。

10.12 ---(MCRA) マスタコントロールリレーの開始

MCR ファンクションの使用方法に関する重要事項

シンボル

---(MCRA)

説明

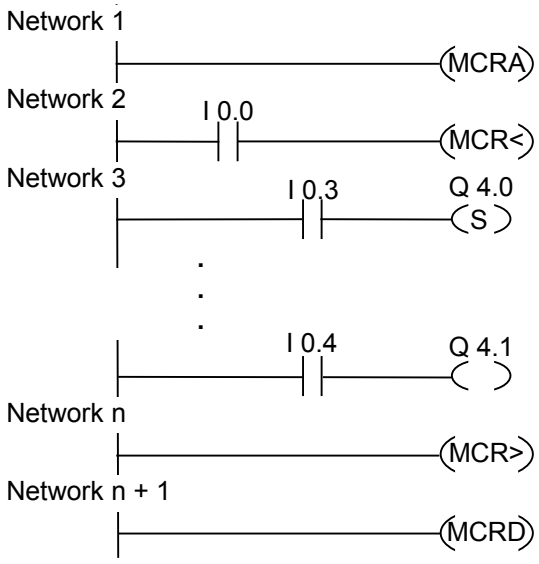
---(MCRA) (マスタコントロールリレーの開始)は、マスタコントロールリレーファンクションを有効にします。このコマンドの実行後、次のコマンドを使って MCR ゾーンをプログラミングすることができます。

- ---(MCR<)
- ---(MCR>)

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	-	-	-	-

例



MCR の機能は、MCRA 回路によって有効になります。MCR<と MCR>間の回路(出力 Q4.0、Q4.1)は、次のように実行されます。

I0.0 = "1" (MCR はオン): I0.3 の論理状態が"1"の場合は Q4.0 が"1"に設定されます。I0.3 が"0"で、I0.4 の論理状態が Q4.1 に割り付けられている場合は、Q4.0 は変わりません。

I0.0 = "0" (MCR はオフ): Q4.0 は、I0.3 の論理状態に関わらず変わりません。Q4.1 は、I0.4 の論理状態に関わらず"0"になります。

次の回路では、命令---(MCRD)により MCR が終了します。つまり、---(MCR<)と---(MCR>)の 2 つの命令による MCR ゾーンのプログラミングはできなくなります。

10.13 ---(MCRD) マスタコントロールリレーの終了

MCR ファンクションの使用方法に関する重要事項

シンボル

---(MCRD)

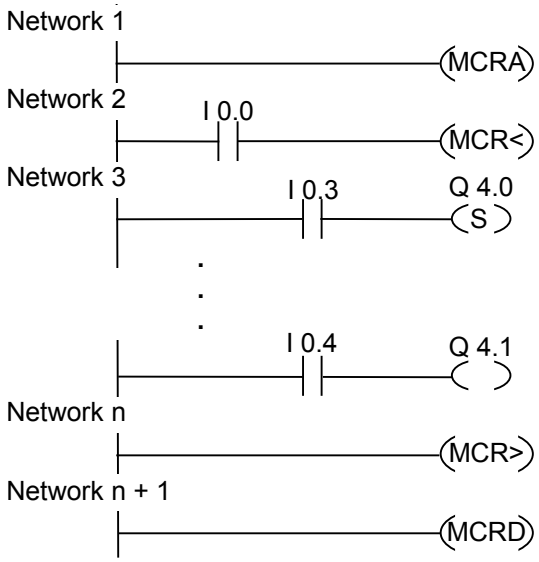
説明

---(MCRD) (マスタコントロールリレーの終了)は、MCR の機能を無効にします。このコマンドの実行後、MCR ゾーンのプログラミングはできなくなります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	-	-	-	-

例



MCR の機能は、MCRA 回路によって有効になります。MCR<と MCR>間の回路(出力 Q4.0、Q4.1)は、次のように実行されます。

I0.0 = "1" (MCR はオン): I0.3 の論理状態が"1"で、I0.4 の論理状態が Q4.1 に割り付けられている場合は、Q4.0 が"1"に設定されます。

I0.0 = "0" (MCR はオフ): Q4.0 は、I0.3 の論理状態に関わらず変わりません。Q4.1 は、I0.4 の論理状態に関わらず"0"になります。

次の回路では、命令---(MCRD)により MCR が終了します。つまり、---(MCR<)と---(MCR>)の 2 つの命令による MCR ゾーンのプログラミングはできなくなります。

10.14 ---(RET) リターン

シンボル

---(RET)

説明

RET(リターン)は、条件付きでブロックを終了する場合に使用します。この処理を行う前に、論理演算が必要です。

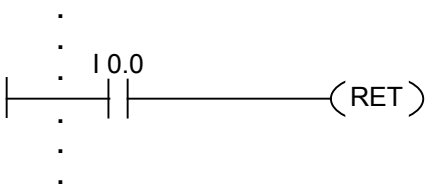
ステータスワード

条件付きリターン(RLO = "1"の場合にリターンを実行):

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	*	-	-	-	0	0	1	1	0

* **RET** 操作は、"SAVE; BEC, "シーケンスで内部的に示されます。これは BR ビットにも影響します。

例



I0.0 が"1"のときに、ブロックが終了します。

11 シフト命令および回転命令

11.1 シフト命令

11.1.1 シフト命令の概要

説明

シフト命令を使用すれば、入力 IN の内容を左または右に 1 ビットずつ移動することができます(「CPU レジスタ」も参照)。左に移動すると、入力 IN の内容が 2 の n 乗(2^n)で乗算されます。右に移動すると、入力 IN の内容が 2 の n 乗(2^n)で除算されます。たとえば、10 進数値 3 の 2 進数表現を左に 3 ビットシフトすると、アキュムレータには 10 進数値 24 の 2 進数表現が得られます。10 進数値 16 の 2 進数表現を右に 2 ビットシフトすると、アキュムレータには 10 進数値 4 の 2 進数表現が得られます。

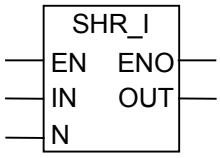
入力パラメータ N に入力する値は、シフトさせるビット数を示します。シフト命令によって空になってしまったビットの桁は 0 になるか符号ビット(0＝正、1＝負)に変わります。最後にシフトしたビットの信号状態は、ステータスワードの CC 1 ビットにロードされます。ステータスワードの CC 0 および OV ビットは 0 にリセットされます。ジャンプ命令を使用して、CC 1 ビットを評価することができます。

使用可能なシフト命令を次に示します。

- SHR_I　整数右シフト
- SHR_DI　倍長整数右シフト
- SHL_W　ワード左シフト
- SHR_W　ワード右シフト
- SHL_DW ダブルワード左シフト
- SHR_DW ダブルワード右シフト

11.1.2 SHR_I 整数右シフト

シンボル

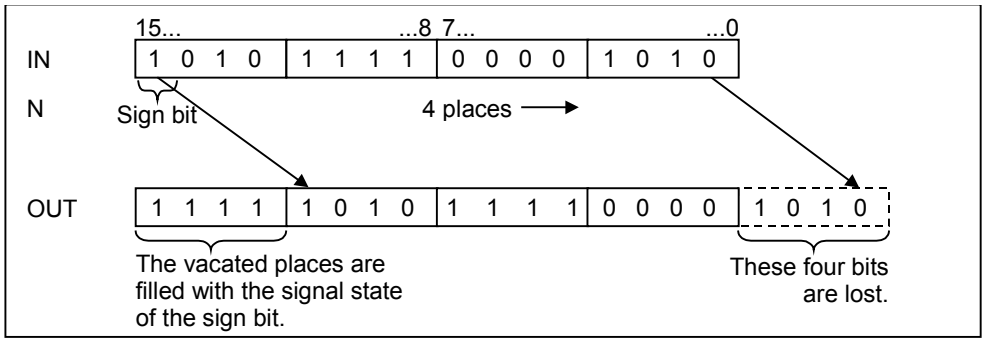


パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	INT	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	INT	I、Q、M、L、D	シフト命令の実行結果

説明

SHR_I (整数右シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHR_I 命令は、入力 IN のビット 0～15 を 1 ビットずつ右へシフトする場合に使用します。ビット 16～31 は変化しません。入力 N では、シフトさせるビット数を指定します。N が 16 より大きい場合、コマンドは、N が 16 であるかのように機能します。空のビット位置を埋めるために、左からシフトしたビット位置には、ビット 15(整数のサインビット)のロジック状態が割り付けられます。つまり、これらのビット位置には、整数が正数の場合は"0"が割り付けられ、負数の場合は"1"が割り付けられます。シフト命令の実行結果は、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

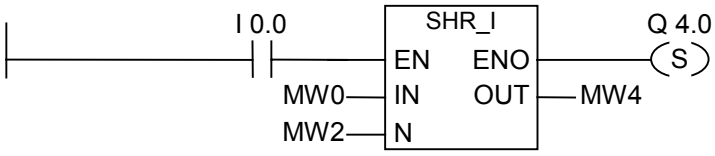
ENO は、EN と同じ信号状態になります。



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

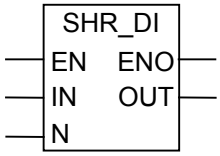
例



SHR_I ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MW0 はロードされ、MW2 に指定されたビット数だけ右にシフトされます。この結果は、MW4 に書き込まれます。Q4.0 は設定されます。

11.1.3 SHR_DI 倍長整数右シフト

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DINT	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	DINT	I、Q、M、L、D	シフト命令の実行結果

説明

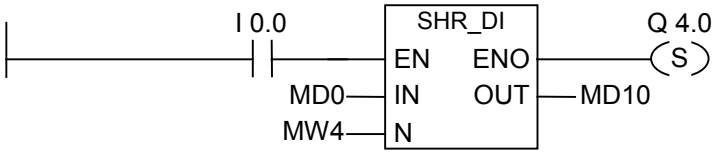
SHR_DI (倍長整数右シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHR_DI 命令は、入力 IN のビット 0～31 を 1 ビットずつ右へシフトします。入力 N では、シフトさせるビット数を指定します。N が 32 より大きい場合、コマンドは、N が 32 であるかのように機能します。空のビット位置を埋めるために、左からシフトしたビット位置には、ビット 31(倍数整数のサインビット)のロジック状態が割り付けられます。つまり、これらのビット位置には、整数が正数の場合は"0"が割り付けられ、負数の場合は"1"が割り付けられます。シフト命令の実行結果は、出力 OUT で確認できます。N が 0 以外の場合は、CC0 ビットと OV ビットは"0"に設定されます。

ENO は、EN と同じ信号状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

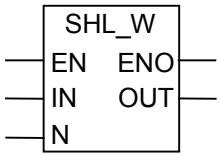
例



SHR_DI ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MD0 はロードされ、MW4 に指定されたビット数だけ右にシフトされます。この結果は、MD10 に書き込まれます。Q4.0 は設定されます。

11.1.4 SHL_W ワード左シフト

シンボル

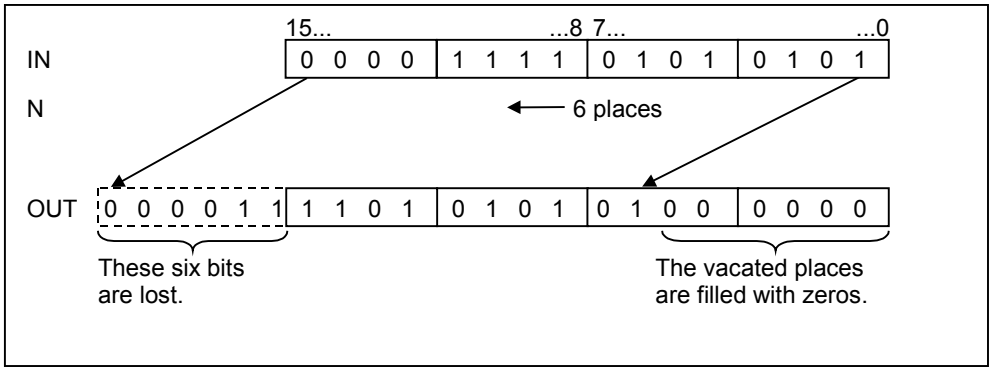


パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	WORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	WORD	I、Q、M、L、D	シフト命令の実行結果

説明

SHL_W (ワード左シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHL_W 命令は、入力 IN のビット 0～15 を 1 ビットずつ左へシフトする場合に使用します。ビット 16～31 は変化しません。入力 N では、シフトさせるビット数を指定します。N が 16 より大きい場合、出力 OUT に"0"が書き込まれ、ステータスワードの CC0 ビットと OV ビットは"0"に設定されます。また、N 個のゼロが右側からシフトされ、空いたビット位置が埋められます。シフト命令の実行結果は、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

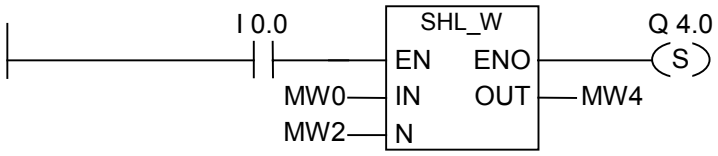
ENO は、EN と同じ信号状態になります。



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

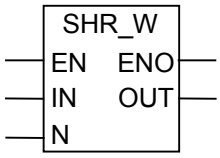
例



SHL_W ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MW0 はロードされ、MW2 に指定されたビット数だけ左にシフトされます。この結果は、MW4 に書き込まれます。Q4.0 は設定されます。

11.1.5 SHR_W ワード右シフト

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	WORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	WORD	I、Q、M、L、D	シフト命令の実行結果(ワード)

説明

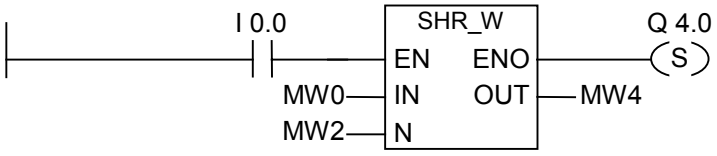
SHR_W (ワード右シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHR_W 命令は、入力 IN のビット 0～15 を 1 ビットずつ右へシフトする場合に使用します。ビット 16～31 は変化しません。入力 N では、シフトさせるビット数を指定します。N が 16 より大きい場合、出力 OUT に"0"が書き込まれ、ステータスワードの CC0 ビットと OV ビットは"0"に設定されます。また、N 個のゼロが左側からシフトされ、空いたビット位置が埋められます。シフト命令の実行結果は、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

ENO は、EN と同じ信号状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

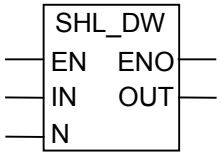
例



SHR_W ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MW0 はロードされ、MW2 に指定されたビット数だけ右にシフトされます。この結果は、MW4 に書き込まれます。Q4.0 は設定されます。

11.1.6 SHL_DW ダブルワード左シフト

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DWORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	DWORD	I、Q、M、L、D	シフト命令の実行結果(ダブルワード)

説明

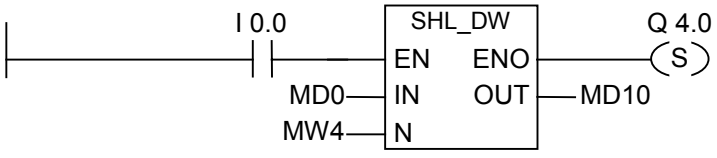
SHL_DW (ダブルワード左シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHL_DW 命令は、入力 IN のビット 0～31 を 1 ビットずつ左へシフトする場合に使用します。入力 N では、シフトさせるビット数を指定します。N が 32 より大きい場合、出力 OUT に"0"が書き込まれ、ステータスワードの CC0 ビットと OV ビットは"0"に設定されます。また、N 個のゼロが右側からシフトされ、空いたビット位置が埋められます。シフト命令の実行結果はダブルワードで出力され、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

ENO は、EN と同じ信号状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

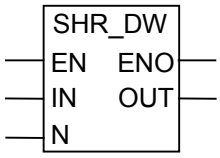
例



SHL_DW ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MD0 はロードされ、MW4 に指定されたビット数だけ左にシフトされます。この結果は、MD10 に書き込まれます。Q4.0 は設定されます。

11.1.7 SHR_DW ダブルワード右シフト

シンボル

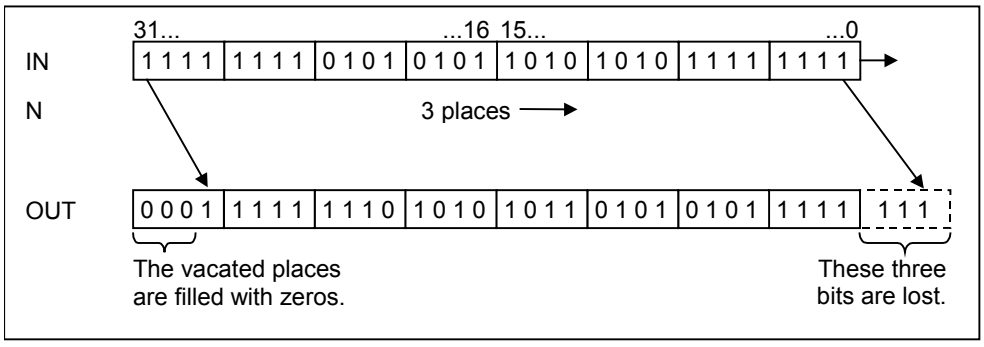


パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DWORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	DWORD	I、Q、M、L、D	シフト命令の実行結果(ダブルワード)

説明

SHR_DW (ダブルワード右シフト)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。SHL_DW 命令は、入力 IN のビット 0～31 を 1 ビットずつ右へシフトします。入力 N では、シフトさせるビット数を指定します。N が 32 より大きい場合、出力 OUT に"0"が書き込まれ、ステータスワードの CC0 ビットと OV ビットは"0"に設定されます。また、N 個のゼロが左側からシフトされ、空いたビット位置が埋められます。シフト命令の実行結果はダブルワードで出力され、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

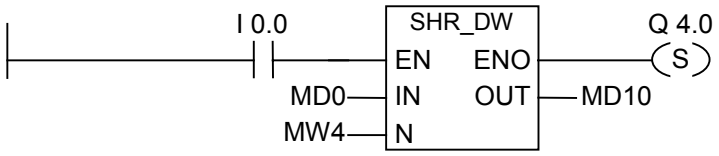
ENO は、EN と同じ信号状態になります。



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

例



SHR_DW ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MD0 はロードされ、MW4 に指定されたビット数だけ右にシフトされます。この結果は、MD10 に書き込まれます。Q4.0 は設定されます。

11.2 回転命令

11.2.1 回転命令の概要

説明

回転命令を使用すると、入力 IN の内容全体を左または右へ 1 ビットずつ移動できます。回転によって空いたビットの桁には、押し出されたビットが入り、結果的にビットの中身が回転します。

入力パラメータ N に入力する値は、回転させるビット数を示します。

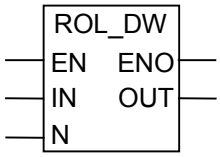
命令に応じて、ステータスワードの CC 1 ビットを使用して回転が行われます。ステータスワードの CC 0 ビットは 0 にリセットされます。

使用可能な回転命令を次に示します。

- ROL_DW　　ダブルワード左回転
- ROR_DW　　ダブルワード右回転

11.2.2 ROL_DW ダブルワード左回転

シンボル

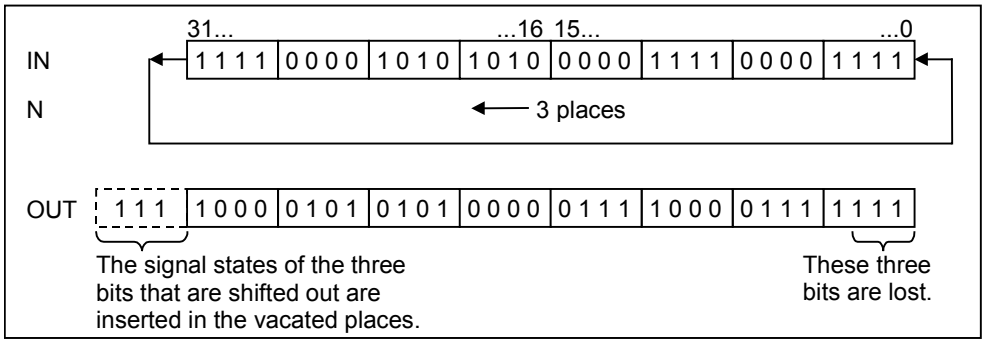


パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DWORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	DWORD	I、Q、M、L、D	シフト命令の実行結果(ダブルワード)

説明

ROL_DW (ダブルワード左回転)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。ROL_DW 命令は、入力 IN の内容全体を 1 ビットずつ左へ回転させる場合に使用します。入力 N では、回転させるビット数を指定します。N が 32 を超える場合は、ダブルワードの位置で移動が行われます((N-1)モジュロ 32) +1。右から移動したビット位置は、左へ循環したビットの論理状態に割り付けられます。実行結果はダブルワードで出力され、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

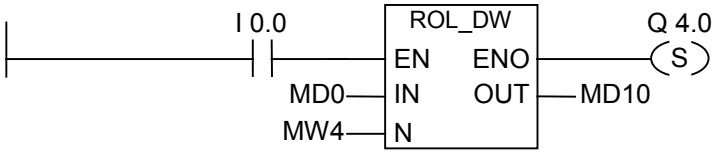
ENO は、EN と同じ信号状態になります。



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

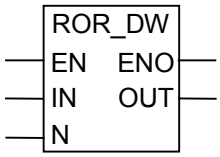
例



ROL_DW ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MD0 はロードされ、MW4 に指定されたビット数だけ左に循環します。この結果は、MD10 に書き込まれます。Q4.0 は設定されます。

11.2.3 ROR_DW ダブルワード右回転

シンボル

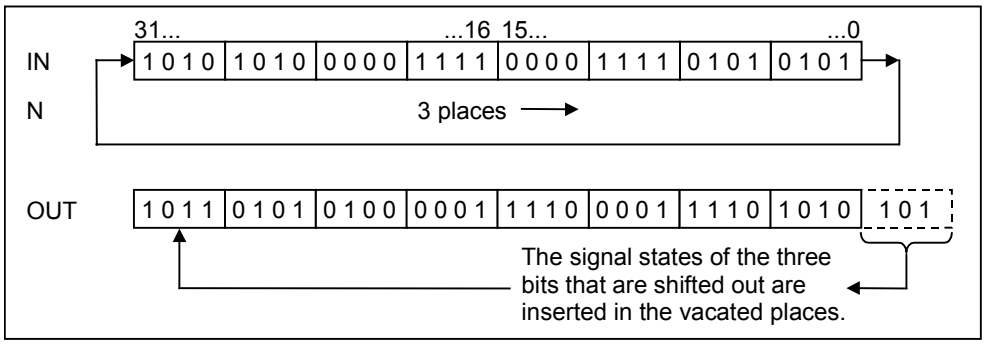


パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN	DWORD	I、Q、M、L、D	シフトさせる値
N	WORD	I、Q、M、L、D	シフトさせるビット位置の数
OUT	DWORD	I、Q、M、L、D	シフト命令の実行結果(ダブルワード)

説明

ROR_DW (ダブルワード右回転)は、入力の有効化(EN)のロジック状態が"1"のときに起動します。ROR_DW 命令は、入力 IN の内容全体を 1 ビットずつ右へ回転させる場合に使用します。入力 N では、回転させるビット数を指定します。N が 32 を超える場合は、ダブルワードの位置で移動が行われます((N-1)モジュロ 32) +1。左から移動したビット位置は、右へ循環したビットの論理状態に割り付けられます。実行結果はダブルワードで出力され、出力 OUT で確認できます。N が 0 以外の場合、CC0 ビットと OV ビットは"0"に設定されます。

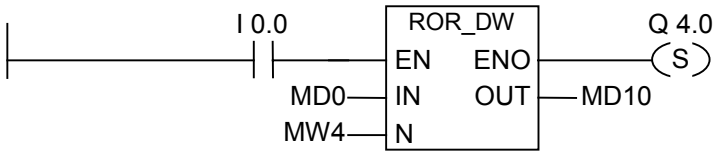
ENO は、EN と同じ信号状態になります。



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	x	x	x	x	-	x	x	x	1

例



ROR_DW ボックスは、I0.0 のロジック"1"によってアクティブ化されます。MD0 はロードされ、MW4 に指定されたビット数だけ右に循環します。この結果は、MD10 に書き込まれます。Q4.0 は設定されます。

12 ステータスビット命令

12.1 ステータスビット命令の概要

説明

ステータスビット命令は、ステータスワードのビットと連動するビット論理命令です。これらの各命令は、次の条件のいずれかに反応します。これらの条件は、ステータスワードの 1 つ以上のビットによって示されます。

- バイナリリザルトビット (BR ---I I---)が設定される(信号状態が 1 になる)。
- 演算ファンクションに オーバーフロー (OV ---I I---)またはストアドオーバーフロー(OS ---I I---)がある。
- 演算ファンクションの結果が無効 (UO ---I I---)である。
- 演算ファンクションの結果が、次のいずれかの方法で 0 に関係する。
  == 0、<> 0、> 0、< 0、>= 0、<= 0

ステータスビット命令が連続して接続されている場合は、AND 真理値表に従って、信号状態チェックの結果と前の論理演算の結果が結合されます。ステータスビット命令が並列して接続されている場合は、OR 真理値表に従って、命令の結果と前の RLO が結合されます。

ステータスワード

ステータスワードは、CPU のメモリ内にあるレジスタで、ビットアドレスで参照できるビットとワード論理命令が含まれています。ステータスワードのストラクチャを次に示します。

2^{15}...	...2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
		BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC

ステータスワード内のビットは、次の方法で評価することができます。

- 整数演算ファンクション
- 浮動小数点ファンクション

12.2 OV ---| |--- 例外ビットオーバーフロー

シンボル

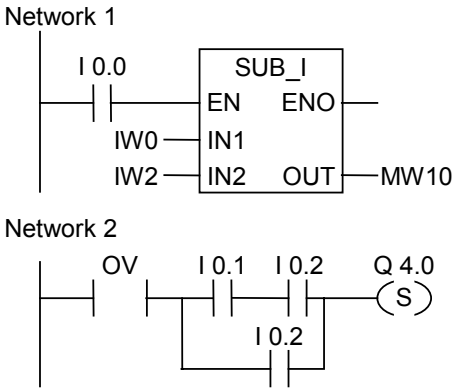
説明

OV ---| |--- (例外ビットオーバーフロー)または **OV ---| / |---** (否定例外ビットオーバーフロー)の接点記号は、直前に実行された演算ファンクションがオーバーフローしたことを示します。つまり、ファンクションの実行後、命令の結果が負から正の有効範囲にないことを表します。この接点を直列接続で使用する場合、スキャン結果は AND により RLO にリンクされ、並列接続で使用する場合は、OR により RLO にリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。数値演算"IW0-IW2"の結果が整数の許容範囲外の場合は、OV ビットが設定されます。

OV の信号状態のスキャンは"1"になります。OV のスキャン結果が信号状態"1"で、ネットワーク 2 の RLO が"1"の場合、Q4.0 が設定されます。

注記

OV のスキャンは、2 つの独立したネットワークがある場合にのみ必要です。それ以外の場合は、演算ファンクションの結果が有効範囲内にない場合、演算ファンクションの ENO 出力が"0"になります。

12.3 OS ---| |--- 例外ビットオーバーフローの保存

シンボル

説明

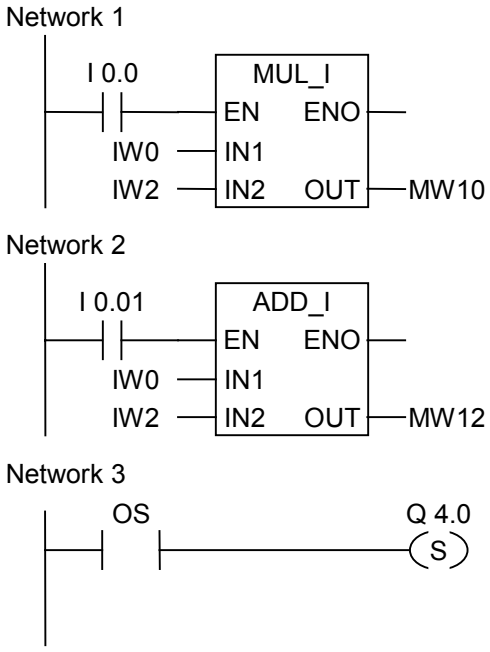
OS ---| |--- (例外ビットオーバーフローの保存)または **OS ---| / |---** (例外ビットオーバーフローの保存の否定)の接点記号は、演算ファンクションのラッチングオーバーフローを示し、それを保存します。命令の結果が負から正の有効範囲にない場合、ステータスワードの OS ビットが設定されます。以降の演算ファンクションに応じて書き直される OV ビットとは異なり、OS ビットは、オーバーフローが発生すると、そのオーバーフローを保存します。OS ビットは、ブロックが残っている限り設定されたままになります。

この接点を直列接続で使用する場合、スキャン結果は AND により RLO にリンクされ、並列接続で使用する場合は、OR により RLO にリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



MUL_I ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。ADD_I ボックスは、I0.1 のロジック"1"によってアクティブ化されます。いずれかの数値演算の結果が、整数の有効範囲にない場合、ステータスワードの OS ビットが"1"に設定されます。OS のスキャンがロジック"1"の場合、Q4.0 が設定されます。

注記

OS のスキャンは、2 つの独立したネットワークがある場合にのみ必要です。それ以外の場合は、最初の数値演算の ENO 出力を 2 番目の演算の EN 入力に接続することが可能です(カスケード配置)。

12.4 UO ---| |--- 例外ビット誤り検出

シンボル

説明

UO ---| |--- (例外ビット誤り検出)または **UO ---| / |---** (例外ビット誤り検出の否定)の接点記号は、浮動小数点数を使用する演算ファンクションに誤り(演算ファンクションに無効な浮動小数点数がある)があるかどうかを示します。

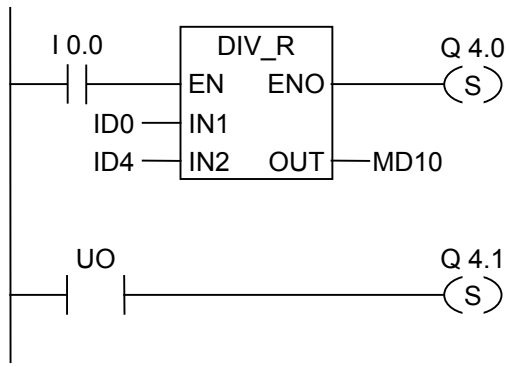
浮動小数点数(UO)を含む数値の結果が無効の場合、信号状態のスキャンは"1"になります。CC1 と CC0 の論理演算で"無効ではない"という結果が出た場合、信号状態のスキャン結果は"0"になります。

この接点を直列接続で使用する場合、スキャン結果は AND により RLO にリンクされ、並列接続で使用する場合は、OR により RLO にリンクされます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。ID0 または ID4 の値が無効な浮動小数点数の場合、数値演算は無効になります。EN の信号状態が 1(有効)の場合、およびファンクション DIV_R の処理中にエラーが発生した場合、ENO の信号状態は 0 になります。

ファンクション DIV_R の実行時に、無効な**浮動小数点数**が含まれている場合、出力 Q4.1 が設定されます。

12.5 BR ---| |--- 例外ビットバイナリリザルト

シンボル

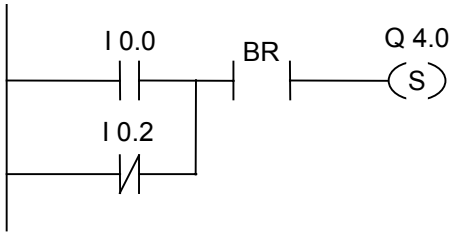
説明

BR ---| |--- (例外ビットBRメモリ)または**BR ---| / |---**(例外ビットBRメモリの否定)の接点記号は、ステータスワードの BR ビットの論理状態をテストします。この接点を直列接続で使用する場合、スキャン結果は AND により RLO にリンクされ、並列接続で使用する場合は、OR により RLO にリンクされます。BR ビットは、ワード処理からビット処理への移行に使用されます。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



I0.0 が"1"または I0.2 が"0"の場合に Q4.0 が設定され、さらに、BR ビットの論理状態が"1"になります。

12.6 ==0 ---| |--- リザルトビット(0)

シンボル

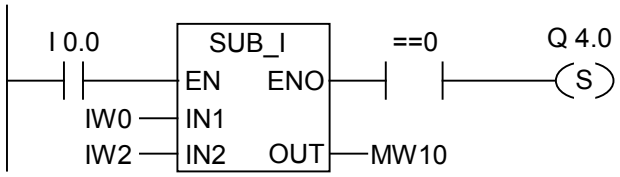
説明

==0 ---| |--- (0 と等しいリザルトビット) または**==0 ---| / |---** 0(0 と等しいリザルトビットの否定)接点記号は、数値演算の結果が 0 と等しいかどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

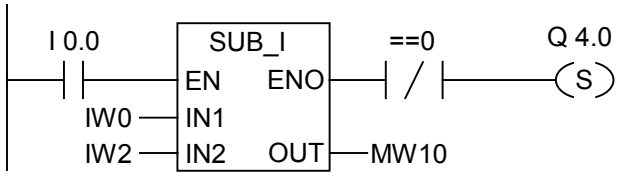
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値と等しい場合、演算ファンクション IW0-IW2 の結果は"0"になります。ファンクションが正しく実行され、結果が"0"と**等しい**場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が**"0"でない**場合、Q4.0 が設定されます。

12.7 <>0 ---| |--- リザルトビット(0 以外)

シンボル

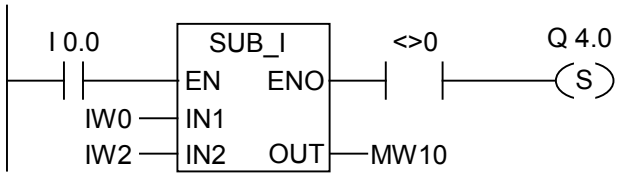
説明

<>0 ---| |--- (0 と等しくないリザルトビット) または**<> ---| / |---** 0(0 と等しくないリザルトビットの否定)接点記号は、数値演算の結果が 0 と等しくないかどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

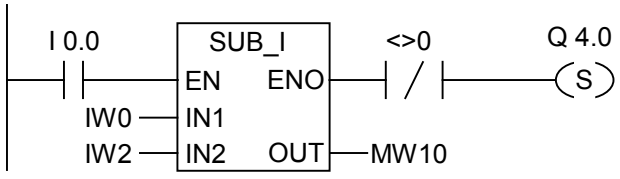
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値と異なる場合、演算ファンクション IW0-IW2 の結果は"0"ではない値になります。ファンクションが正しく実行され、結果が"0"と**等しくない**場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が**"0"**の場合、Q4.0 が設定されます。

12.8 >0 ---| |--- リザルトビット>0

シンボル

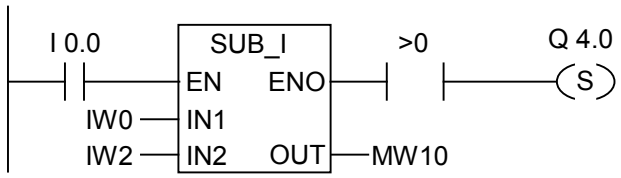
説明

>0 ---| |--- (0 より大きいリザルトビット) または **>0 ---| / |---** (0 より大きいリザルトビットの否定) 接点記号は、数値演算の結果が 0 より大きいかどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

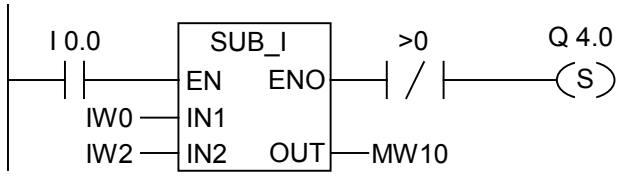
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値より大きい場合、演算ファンクション IW0-IW2 の結果は"0"より大きくなります。ファンクションが正しく実行され、結果が"0"より**大きい**場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が"0"**以下**の場合、Q4.0 が設定されます。

12.9 <0 ---| |--- リザルトビット<0

シンボル

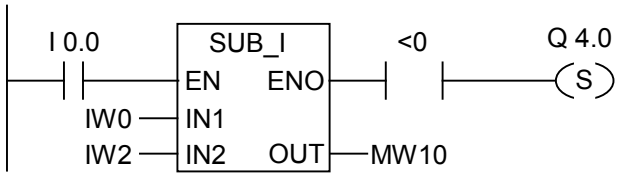
説明

<0 ---| |--- (0 より小さいリザルトビット) または **<0 ---| / |---** 0(0 より小さいリザルトビットの否定) 接点記号は、数値演算の結果が 0 より小さいかどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

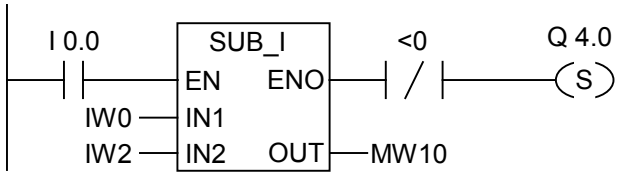
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値より小さい場合、演算ファンクション IW0-IW2 の結果は"0"より小さくなります。ファンクションが正しく実行され、結果が"0"より**小さい**場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が"0"**以上**の場合、Q4.0 が設定されます。

12.10 >=0 ---| |--- リザルトビット(0 以上)

シンボル

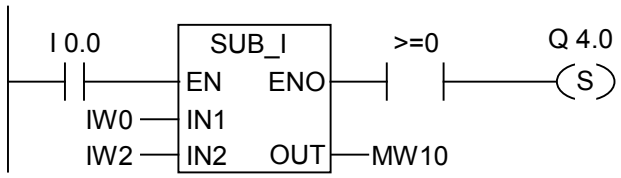
説明

>=0 ---| |--- (0 以上のリザルトビット) または **>=0 ---| / |---** 0(0 以上のリザルトビットの否定)接点記号は、数値演算の結果が 0 以上かどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

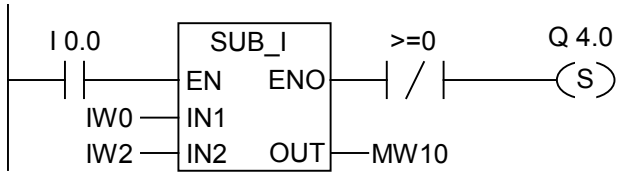
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値以上の場合、演算ファンクション IW0-IW2 の結果は"0"以上になります。ファンクションが正しく実行され、結果が"0"**以上**の場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が"0"**以下**の場合、Q4.0 が設定されます。

12.11 <=0 ---| |--- リザルトビット(0 以下)

シンボル

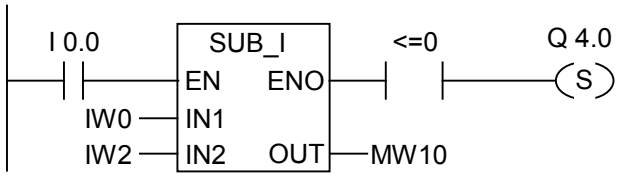
説明

<=0 ---| |--- (0 以下のリザルトビット) または **<=0 ---| / |---** 0(0 以下のリザルトビットの否定)接点記号は、数値演算の結果が 0 以下かどうかを認識するために使用します。これらの命令は、ステータスワードの条件コードビット CC1 および CC0 をスキャンし、"0"との関係を判定します。直列で使用すると、スキャン結果は RLO に AND でリンクされます。並列で使用すると、スキャン結果は RLO に OR でリンクされます。

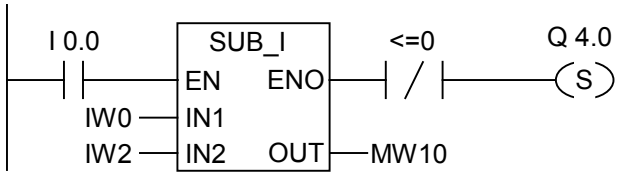
ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



ボックスは、I0.0 の信号状態"1"によってアクティブ化されます。IW0 の値が IW2 の値以下の場合、演算ファンクション IW0-IW2 の結果は"0"以下になります。ファンクションが正しく実行され、結果が"0"**以下**の場合、Q4.0 が設定されます。



演算ファンクションが正しく実行され、その結果が"0"**以上**の場合、Q4.0 が設定されます。

13 タイマ命令

13.1 タイマ命令の概要

説明

- 正しい時間の設定と選択については、「メモリ内のタイマの場所およびタイマのコンポーネント」を参照してください。

使用可能なタイマ命令を次に示します。

- S_PULSE パルス S5 タイマ
- S_PEXT 拡張パルス S5 タイマ
- S_ODT オンディレイ S5 タイマ
- S_ODTS 拡張オンディレイ S5 タイマ
- S_OFFDT オフディレイ S5 タイマ
- ---(SP) パルスタイマコイル
- ---(SE) 拡張パルスタイマコイル
- ---(SD) オンディレイタイマコイル
- ---(SS) 拡張オンディレイタイマコイル
- ---(SA) オフディレイタイマコイル

13.2 メモリ内のタイマの場所およびタイマのコンポーネント

メモリ内の領域

タイマには、CPU のメモリ内に専用の領域が割り付けられています。このメモリ領域は、各タイマアドレスに対して 1 つの 16 ビットワードを確保します。ラダーロジック命令セットには 256 個のタイマがあります。使用可能なタイマワードの数を設定する場合には、CPU の技術情報を参照してください。

以下の機能を使って、タイマのメモリ領域へアクセスできます。

- タイマ命令
- クロックタイミングによるタイマワードの更新。この機能は、CPU が RUN モードになっているときに使用可能で、タイマ値が 0 になるまで、指定されたタイムベースで単位時間ずつ設定値を減少します。

時間値

タイマワードのビット 0～9 には、時間値がバイナリコードで格納されます。時間値では、多数の単位が指定されます。時間を更新すると、タイムベースで指定された間隔で時間値が 1 単位ずつ減っていきます。タイマ値は、0 になるまで減少し続けます。タイマ値は、2 進数表記、16 進数表記、または 2 進化 10 進数(BCD)表記で、アキュムレータ 1 の下位ワードへロードできます。

次のいずれかのフォーマットを使用して、時間値を事前にロードすることができます。

- W#16#wxyz
  - w = タイムベース(時間間隔、分解能)
  - xyz = 2 進化 10 進フォーマットの時間値
- S5T#a**H**_b**M**_c**S**_d**MS**
  - ここで 、H = 時間、M = 分、S = 秒、MS = ミリ秒を表します。
    a、b、c、d はユーザー定義です。
  - タイムベースは自動的に選択され、タイマ値は選択されたタイムベースごとに一単位ずつ減少します。

入力できる最大タイマ値は、9,990 秒または 2H_46M_30S です。

S5TIME#4S = 4 秒
s5t#2h_15m = 2 時間 15 分
S5T#1H_12M_18S = 1 時間 12 分 18 秒

タイムベース

タイマワードのビット 12 とビット 13 には、タイムベースがバイナリコードで格納されます。タイムベースは、時間値が一単位ずつ減少する間隔を定義します。最小タイムベースは 10 ms で、最大タイムベースは 10 s です。

タイムベース	タイムベースのバイナリコード
10 ms	00
100 ms	01
1 秒	10
10 秒	11

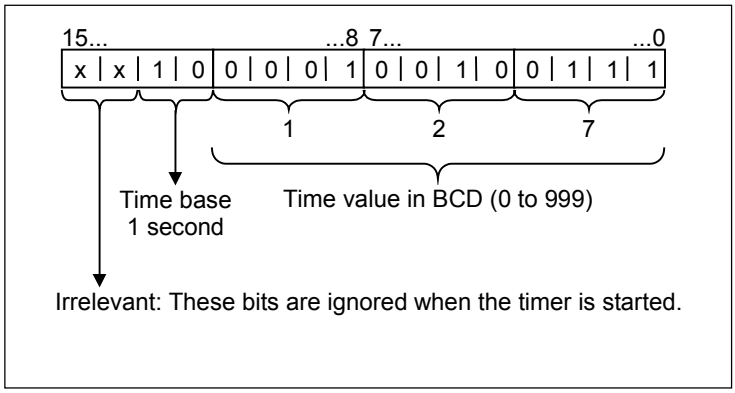
2h46m30s を超える値は無効です。分解能が有効範囲よりも大きい場合(たとえば、2h10ms)、有効範囲になるように、その値に対して切り捨てが実行されます。S5TIME の一般形式では、値の範囲と分解能について次のような制限があります。

分解能	範囲
0.01 秒	10MS ～ 9S_990MS
0.1 秒	100MS～1M_39S_900MS
1 秒	1S～16M_39S
10 秒	10S～2H_46M_30S

タイマセルのビットコンフィグレーション

タイマを起動する場合、タイマセルの内容が時間値として使用されます。タイマセルのビット 0 から 11 は、BCD 表記のタイマ値を保持しています。ビット 12 とビット 13 には、タイムベースがバイナリコードで格納されます。

下図は、タイマ値を 127 とし、タイムベースを 1 秒としてロードした場合のタイマセルの内容を表したものです。

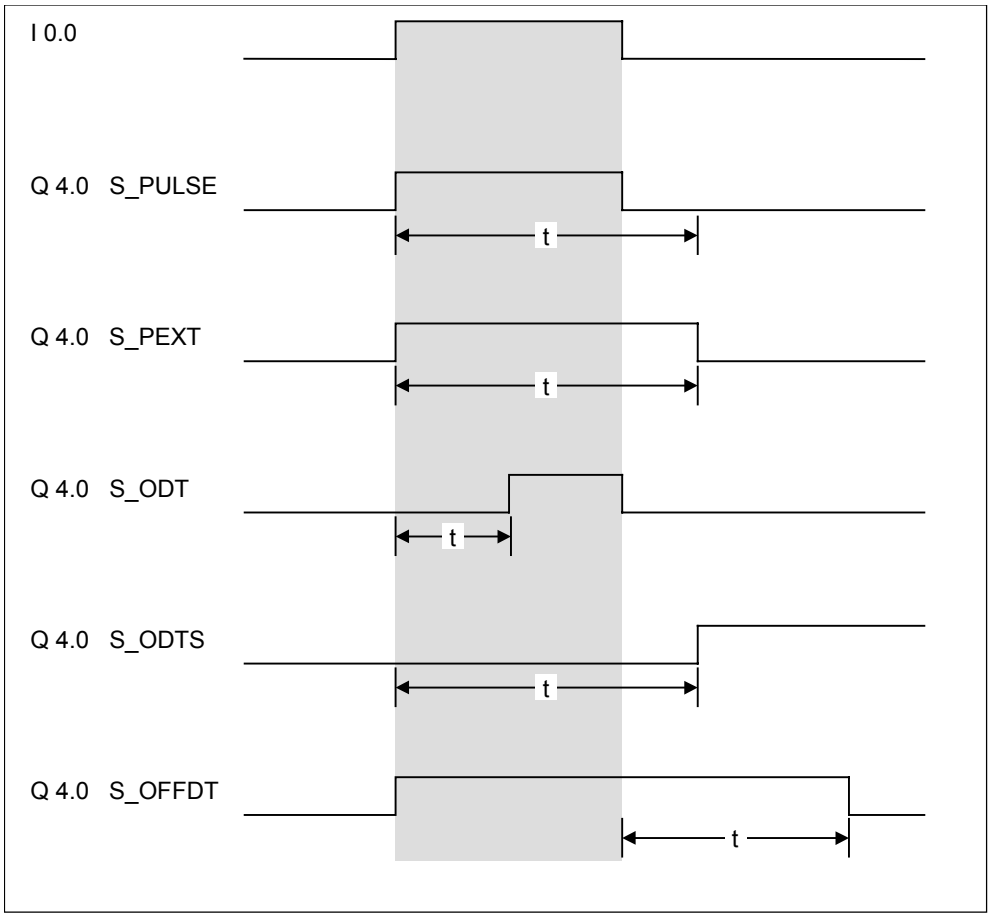


時間値とタイムベースの読み取り

1 つのタイマボックスに対し、BI と BCD の 2 つの出力があり、この 2 つの出力にワード位置を指定できます。BI 出力では、時間値がバイナリフォーマットで示されます。BCD 出力では、タイムベースと時間値が 2 進化 10 進(BCD)フォーマットで示されます。

適切なタイマの選択

この概要を参考にすれば、タイミングジョブに適切なタイマを選択できます。

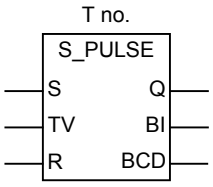


タイマ	説明
S_PULSE パルスタイマ	出力信号が 1 となる最大時間は、プログラムされたタイマ値 t と同じです。入力信号が 0 に変わる場合は、出力信号が短い間だけ 1 になります。
S_PEXT 拡張パルスタイマ	入力信号が 1 である時間に関係なくプログラムされた時間 t の間、出力信号は 1 です。
S_ODT オンディレイタイマ	プログラムされた時間 t が経過し、入力信号が 1 である場合にだけ、出力信号が 1 に変わります。
S_ODTS 拡張オンディレイタイマ	プログラムされた時間 t が経過すると、入力信号が 0 のままかどうかに関係なく、出力信号が 1 になります。
S_OFFDT オフディレイタイマ	入力信号が 1 に変わる場合、またはタイマが作動している間、出力信号は 1 になります。入力信号が 1 から 0 に変わると、タイマが開始します。

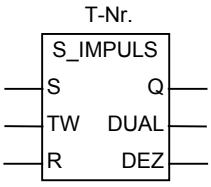
13.3 S_PULSE パルス S5 タイマ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
T no.	T-Nr.	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
S	S	BOOL	I、Q、M、L、D	起動入力
TV	TW	S5TIME	I、Q、M、L、D	設定済みの時間
R	R	BOOL	I、Q、M、L、D	リセット入力
BI	DUAL	WORD	I、Q、M、L、D	残り時間の値(整数値)
BCD	DEZ	WORD	I、Q、M、L、D	残り時間の値(BCD フォーマット)
Q	Q	BOOL	I、Q、M、L、D	タイマのステータス

説明

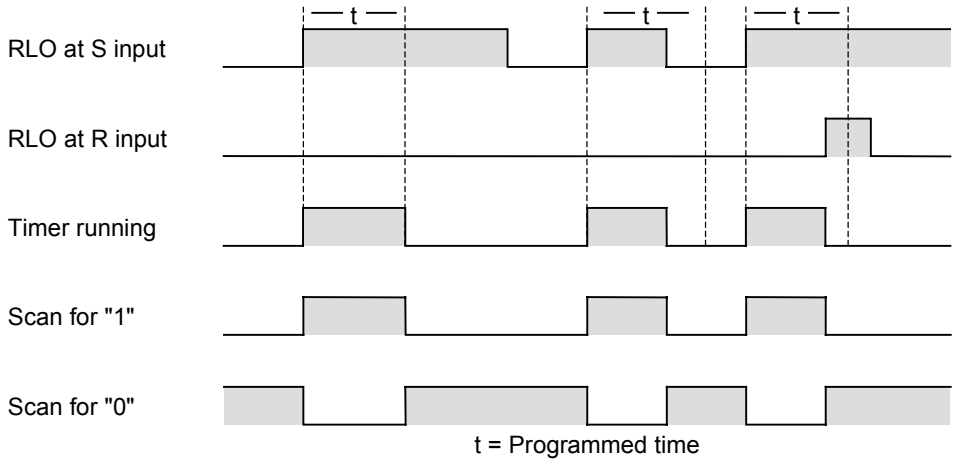
S_PULSE (パルス S5 タイマ)は、スタート(S)入力が信号立ち上がりのときに、指定された時間のカウントを開始します。タイマを起動するには、信号状態の変化が必要です。タイマは、入力 S の信号状態が"1"のときに実行しますが、その実行時間は、入力 TV で指定されているタイマ値までになります。タイマの実行中、出力 Q の信号状態は"1"になります。時間間隔に達する前に S 入力の信号状態が"1"から"0"に変わると、タイマは停止します。この場合、出力 Q の信号状態は"0"になります。

タイマの実行中にタイマリセット(R)入力が"0"から"1"に変わると、タイマはリセットされます。現在の時間およびタイムベースも 0 に設定されます。タイマが実行していない場合は、タイマの R 入力のロジック状態は"1"のままです。

現在の時間は、出力 BI および BCD で確認できます。BI の時間値はバイナリ形式で、BCD の時間値は BCD 形式で表示されます。現在の時間は、TV の初期値から、タイマ開始後の経過時間を引いたものです。

タイムチャート

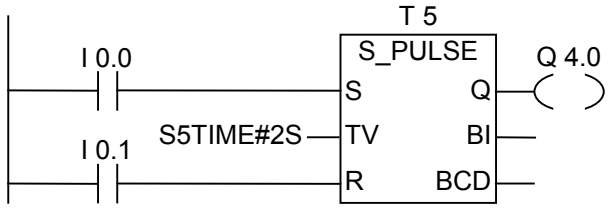
パルスタイマの特性



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



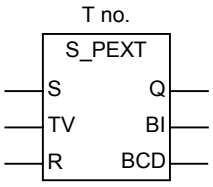
入力 I0.0 の信号状態が"0"から"1"(RLO の信号立ち上がり)に変わると、タイマ T5 が起動します。入力 I0.0 が"1"である場合、タイマは指定された 2 秒間実行を継続します。時間が経過する前に、入力 I0.0 の信号状態が"1"から"0"に遷移した場合、タイマは停止されます。タイマの実行中に入力 I0.1 の信号状態が"0"から"1"に変わると、タイマはリセットされます。

タイマの実行中、出力 Q4.0 はロジック"1"になり、タイマが終了するかリセットされると"0"になります。

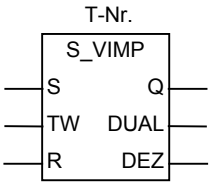
13.4 S_PEXT 拡張パルス S5 タイマ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
T no.	T-Nr.	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
S	S	BOOL	I、Q、M、L、D	起動入力
TV	TW	S5TIME	I、Q、M、L、D	設定済みの時間
R	R	BOOL	I、Q、M、L、D	リセット入力
BI	DUAL	WORD	I、Q、M、L、D	残り時間の値(整数値)
BCD	DEZ	WORD	I、Q、M、L、D	残り時間の値(BCD フォーマット)
Q	Q	BOOL	I、Q、M、L、D	タイマのステータス

説明

S_PEXT (拡張パルス S5 タイマ)は、スタート(S)入力が信号立ち上がりのときに、指定された時間のカウントを開始します。タイマを起動するには、信号状態の変化が必要です。タイマは、時間間隔に達する前に S 入力の信号状態が"0"に変わっても、入力 TV で指定されている時間間隔だけ実行します。タイマの実行中、出力 Q の信号状態は"1"になります。タイマの実行中に入力 S の信号状態が"0"から"1"に変わると、タイマは設定済みの時間を使って再起動("再トリガ")します。

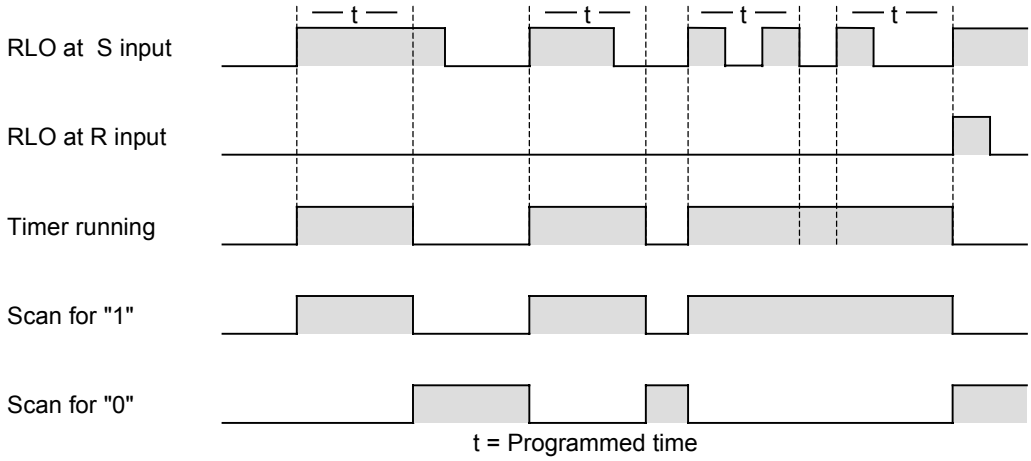
タイマの実行中にリセット(R)入力が"0"から"1"に変わると、タイマはリセットされます。現在の時間およびタイムベースも 0 に設定されます。

現在の時間は、出力 BI および BCD で確認できます。BI の時間値はバイナリ形式で、BCD の時間値は BCD 形式で表示されます。現在の時間は、TV の初期値から、タイマ開始後の経過時間を引いたものです。

関連項目: "メモリ内のタイマの場所およびタイマのコンポーネント"

タイムチャート

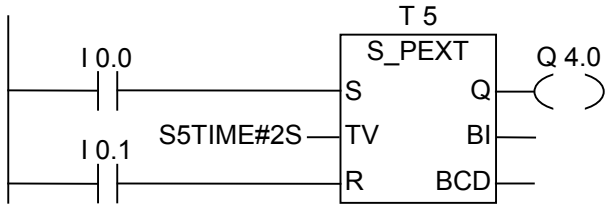
拡張パルスタイマの特性



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例

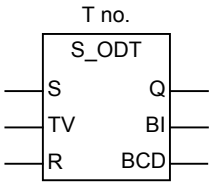


入力 I0.0 の信号状態が"0"から"1"(RLO の信号立ち上がり)に変わると、タイマ T5 が起動します。このタイマは、入力 S の信号立ち下がりに影響されることなく、2 秒(2s)の指定時間だけ実行を継続します。タイマが終了する前に I0.0 の信号状態が"0"から"1"になると、タイマは再トリガされます。タイマの実行中、出力 Q4.0 はロジック"1"になります。

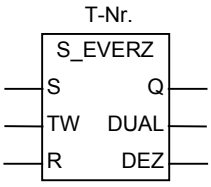
13.5 S_ODT オンディレイ S5 タイマ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
T no.	T-Nr.	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
S	S	BOOL	I、Q、M、L、D	起動入力
TV	TW	S5TIME	I、Q、M、L、D	設定済みの時間
R	R	BOOL	I、Q、M、L、D	リセット入力
BI	DUAL	WORD	I、Q、M、L、D	残り時間の値(整数値)
BCD	DEZ	WORD	I、Q、M、L、D	残り時間の値(BCD フォーマット)
Q	Q	BOOL	I、Q、M、L、D	タイマのステータス

説明

S_ODT (オンディレイ S5 タイマ)は、スタート(S)入力が信号立ち上がりのときに、指定された時間のカウントを開始します。タイマを起動するには、信号状態の変化が必要です。タイマは、入力 S の信号状態が"1"のときに、入力 TV で指定された時間間隔だけ実行します。エラーが発生せずにタイマが終了すると、出力 Q の信号状態は"1"になり、S 入力の信号状態は"1"のままになります。タイマの実行中に、入力 S の信号状態が"1"から"0"に遷移した場合、タイマは停止されます。この場合、出力 Q の信号状態は"0"になります。

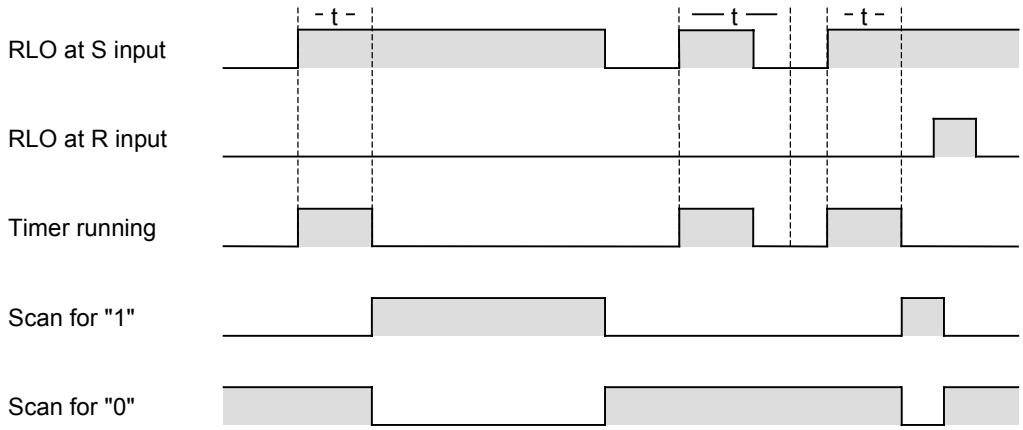
タイマの実行中にリセット(R)入力が"0"から"1"に変わると、タイマはリセットされます。現在の時間およびタイムベースも 0 に設定されます。そうすると、出力 Q の信号状態は"0"になります。タイマが実行されていないときに R 入力のロジックが"1"になった場合や、入力 S の RLO が"1"になった場合もタイマはリセットされます。

現在の時間は、出力 BI および BCD で確認できます。BI の時間値はバイナリ形式で、BCD の時間値は BCD 形式で表示されます。現在の時間は、TV の初期値から、タイマ開始後の経過時間を引いたものです。

関連項目: "メモリ内のタイマの場所およびタイマのコンポーネント"

タイムチャート

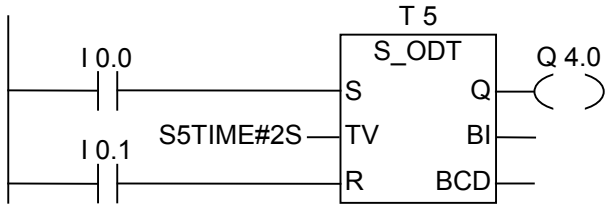
オンディレイタイマの特性



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例

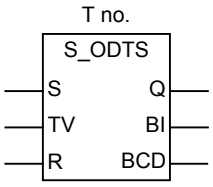


入力 I0.0 の信号状態が"0"から"1"(RLO の信号立ち上がり)に変わると、タイマ T5 が起動します。2 秒が経過しても、入力 I0.0 の信号状態が"1"のままであれば、出力 Q4.0 の信号状態は"1"になります。入力 I0.0 の信号状態が"1"から"0"に遷移した場合、タイマは停止し、出力 Q4.0 は"0"になります。タイマの実行中に入力 I0.1 の信号状態が"0"から"1"に遷移した場合、タイマはリセットされます。

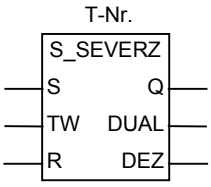
13.6 S_ODTS 拡張オンディレイ S5 タイマ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
T no.	T-Nr.	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
S	S	BOOL	I、Q、M、L、D	起動入力
TV	TW	S5TIME	I、Q、M、L、D	設定済みの時間
R	R	BOOL	I、Q、M、L、D	リセット入力
BI	DUAL	WORD	I、Q、M、L、D	残り時間の値(整数値)
BCD	DEZ	WORD	I、Q、M、L、D	残り時間の値(BCD フォーマット)
Q	Q	BOOL	I、Q、M、L、D	タイマのステータス

説明

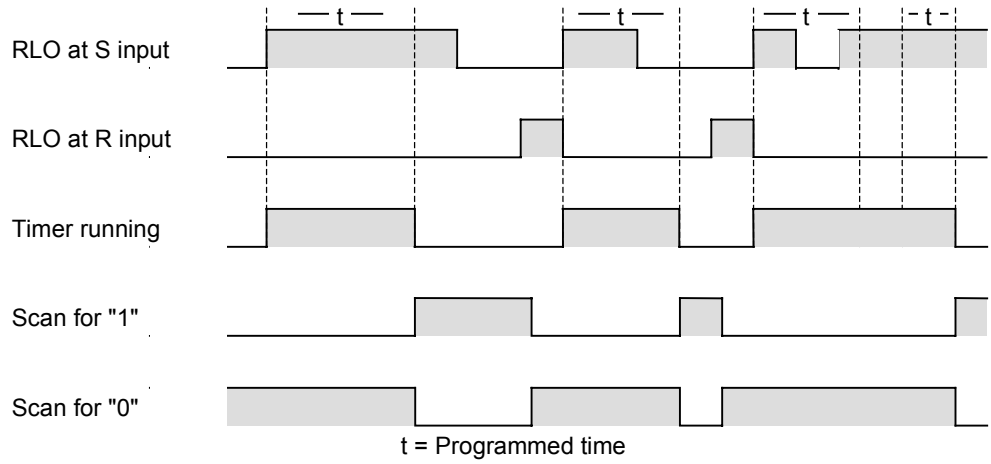
S_ODTS (拡張オンディレイ S5 タイマ)は、スタート(S)入力が信号立ち上がりのときに、指定された時間のカウントを開始します。タイマを起動するには、信号状態の変化が必要です。タイマは、時間間隔に達する前に入力 S の信号状態が"0"に変わっても、入力 TV で指定された時間間隔だけ実行します。入力 S の信号状態に関わらず、タイマが切れると、出力 Q の信号状態が"1"になります。タイマの実行中に入力 S の信号状態が"0"から"1"に変わると、タイマは指定された時間で再起動("再トリガ")します。

S 入力の RLO に関係なく、リセット(R)入力が"0"から"1"になると、タイマはリセットされます。出力 Q の信号状態は"0"になります。

現在の時間は、出力 BI および BCD で確認できます。BI の時間値はバイナリ形式で、BCD の時間値は BCD 形式で表示されます。現在の時間は、TV の初期値から、タイマ開始後の経過時間を引いたものです。

タイムチャート

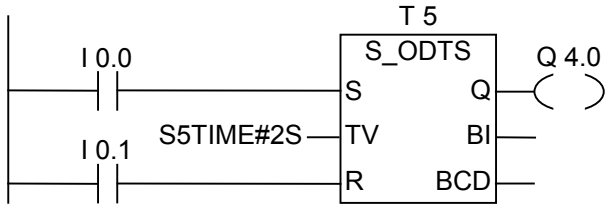
拡張オンディレイタイマの特性



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例

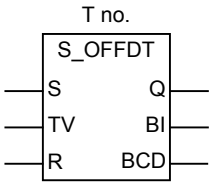


入力 I0.0 の信号状態が"0"から"1"(RLO の信号立ち上がり)に変わると、タイマ T5 が起動します。入力 I0.0 の信号状態が"1"から"0"に遷移するかどうかにかかわらず、タイマは実行されます。タイマが期限切れになる前に入力 I0.0 の信号状態が"0"から"1"に遷移した場合、タイマは再起動されます。タイマが終了すると、出力 Q4.0 は"1"になります。(入力 I0.1 の信号状態が"0"から"1"になると、S の RLO に関係なくタイマはリセットされる)。

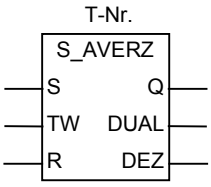
13.7 S_OFFDT オフディレイ S5 タイマ

シンボル

English



German



パラメータ 英語	パラメータ ドイツ語	データタイプ	メモリ領域	説明
T no.	T-Nr.	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
S	S	BOOL	I、Q、M、L、D	起動入力
TV	TW	S5TIME	I、Q、M、L、D	設定済みの時間
R	R	BOOL	I、Q、M、L、D	リセット入力
BI	DUAL	WORD	I、Q、M、L、D	残り時間の値(整数値)
BCD	DEZ	WORD	I、Q、M、L、D	残り時間の値(BCD フォーマット)
Q	Q	BOOL	I、Q、M、L、D	タイマのステータス

説明

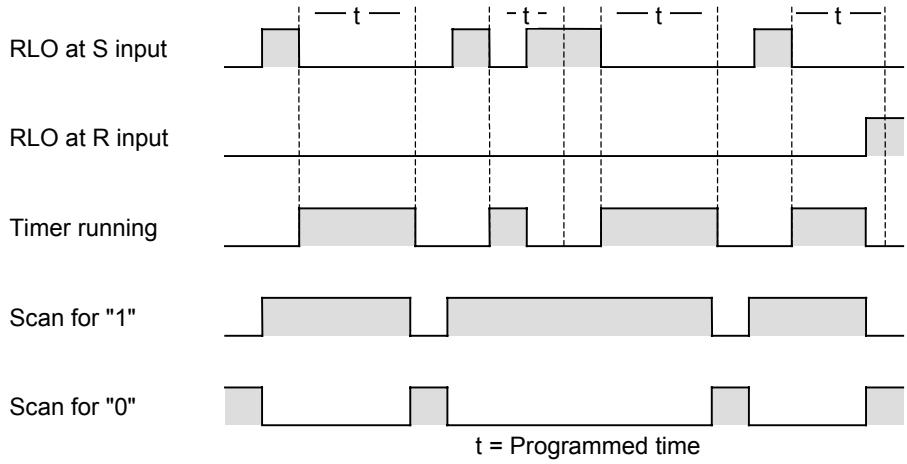
S_OFFDT (オフディレイ S5 タイマ)は、スタート(S)入力が信号立ち下がりのときに、指定された時間のカウントを開始します。タイマを起動するには、信号状態の変化が必要です。入力 S の信号状態が"1"の場合やタイマが実行中の場合、出力 Q の信号状態は"1"になります。タイマの実行中に、入力 S の信号状態が"0"から"1"になると、タイマはリセットされます。入力 S の信号状態が"1"から"0"に戻るまで、タイマは再起動しません。

タイマの実行中にリセット(R)入力が"0"から"1"に変わると、タイマはリセットされます。

現在の時間は、出力 BI および BCD で確認できます。BI の時間値はバイナリ形式で、BCD の時間値は BCD 形式で表示されます。現在の時間は、TV の初期値から、タイマ開始後の経過時間を引いたものです。

タイムチャート

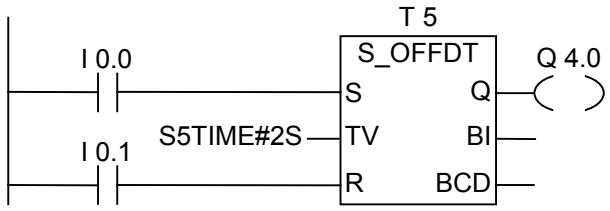
オフディレイタイマの特性



ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	x	x	x	1

例



I0.0 の信号状態が"1"から"0"に変わると、タイマが開始します。

I0.0 が"1"の場合やタイマが実行中の場合、Q4.0 は"1"になります (タイマの実行中に I0.1 の信号状態が"0"から"1"になると、タイマはリセットされる)。

13.8 ---(SP) パルスタイマコイル

シンボル

英語	ドイツ語
<T no..>	<T no.>
---(SP)	**---(SI)**
<time value>	<time value>

パラメータ	データタイプ	メモリ領域	説明
<T no.>	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
<time value>	S5TIME	I、Q、M、L、D	設定済みの時間

説明

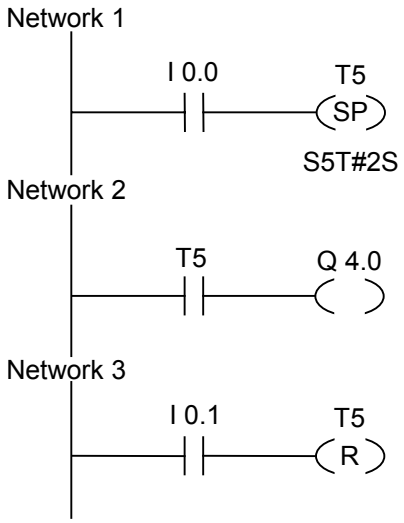
---(SP) (パルスタイマコイル)は、RLO が信号立ち上がりの状態のときに、指定されたタイマを**<time value>**で起動します。RLO が信号立ち上がり("1")のときは、指定された時間間隔だけ、タイマが実行します。タイマの実行中は、カウンタの信号状態は"1"になります。時間値に達する前に RLO が"1"から"0"になると、タイマは停止します。この場合、"1"のスキャンは、常に結果"0"を生成します。

関連項目:「メモリ内のタイマの場所およびタイマのコンポーネント」および「S_PULSE (パルス S5 タイマ)」

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



入力 I0.0 の信号状態が"0"から"1"になると(RLO が信号立ち上がり)、タイマ T5 が起動します。入力 I0.0 の信号状態が"1"である場合、タイマは指定された時間(2 秒間)実行を継続します。指定した時間が経過する前に、入力 I0.0 の信号状態が"1"から"0"に遷移した場合、タイマは停止されます。

タイマの実行中、出力 Q4.0 の信号状態は"1"になります。入力 I0.1 の信号状態が"0"から"1"になると、タイマ T5 はリセットされます。タイマは停止し、タイマ値の残りの部分は"0"にクリアされます。

13.9 ---(SE) 拡張パルスタイマコイル

シンボル

英語	ドイツ語
<T no.>	<T no.>
---(SE)	**---(SV)**
<time value>	<time value>

パラメータ	データタイプ	メモリ領域	説明
<T no.>	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
<time value>	S5TIME	I、Q、M、L、D	設定済みの時間

説明

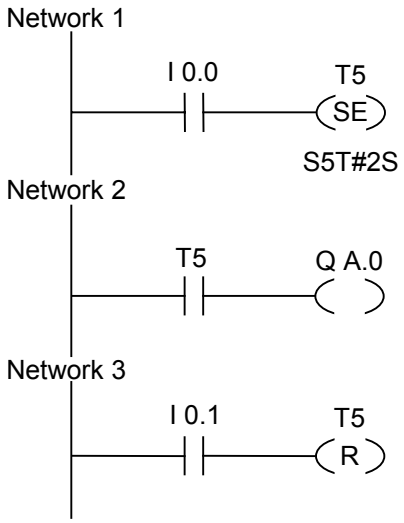
---(SE) (拡張パルスタイマコイル)は、RLO が信号立ち上がりの状態になると、指定された**<time value>**で指定されたタイマを起動します。タイマは、タイマが終了する前に RLO が"0"に変わったとしても、指定された時間間隔だけ実行します。タイマの実行中は、カウンタの信号状態は"1"になります。タイマの実行中に RLO が"0"から"1"に変わると、タイマは、指定された時間値で再起動(再トリガ)されます。

関連項目:「メモリ内のタイマの場所およびタイマのコンポーネント」および「S_PEXT (拡張パルス S5 タイマ)」

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



入力 I0.0 の信号状態が"0"から"1"になると(RLO が信号立ち上がり)、タイマ T5 が起動します。RLO が信号立ち下がりであっても、タイマは実行し続けます。タイマが終了する前に、I0.0 の信号状態が"0"から"1"に変わると、タイマは再トリガされます。

タイマの実行中、出力 Q4.0 の信号状態は"1"になります。入力 I0.1 の信号状態が"0"から"1"になると、タイマ T5 はリセットされます。タイマは停止し、タイマ値の残りの部分は"0"にクリアされます。

13.10 ---(SD) オンディレイタイマコイル

シンボル

英語	ドイツ語
<T no.>	<T no.>
---(SD)	**---(SE)**
<time value>	<time value>

パラメータ	データタイプ	メモリ領域	説明
<T no.>	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
<time value>	S5TIME	I、Q、M、L、D	設定済みの時間

説明

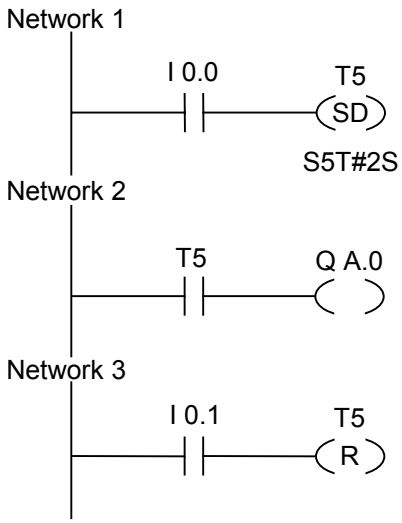
---(SD) (オンディレイタイマコイル)は、RLO が信号立ち上がりのときに、指定されたタイマを**<time value>**で起動します。**<time value>**に達するとタイマの信号状態は"1"になります。このとき、RLO は"1"のままです。タイマの実行中に RLO が"1"から"0"に変わると、タイマはリセットされます。この場合、"1"のスキャンは、常に結果"0"を生成します。

関連項目:「メモリ内のタイマの場所およびタイマのコンポーネント」および「S_ODT (オンディレイ S5 タイマ)」

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



入力 I0.0 の信号状態が"0"から"1"になると(RLO が信号立ち上がり)、タイマ T5 が起動します。指定時間に達し、かつ、入力 I0.0 の信号状態が"1"の場合、出力 Q4.0 の信号状態は"1"になります。

入力 I0.0 の信号状態が"1"から"0"に変わった場合、タイマはアイドルのままで、出力 Q4.0 の信号状態は"0"になります。入力 I0.1 の信号状態が"0"から"1"に変わると、タイマ T5 がリセットされます。これにより、タイマが停止し、残りの時間値がクリアされて"0"になります。

13.11 ---(SS) 拡張オンディレイタイマコイル

シンボル

英語	ドイツ語
<T no.>	<T no.>
---(SS)	**---(SS)**
<time value>	<time value>

パラメータ	データタイプ	メモリ領域	説明
<T no.>	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
<time value>	S5TIME	I、Q、M、L、D	設定済みの時間

説明

---(SS) (拡張オンディレイタイマコイル)は、RLO が信号立ち上がりのときに指定されたタイマを起動します。指定時間に達すると、タイマの信号状態は"1"になります。タイマの再起動は、明示的にリセットした場合にのみ可能です。リセットにより、タイマの信号状態は"0"になります。

タイマの実行中に RLO が"0"から"1"に変わると、指定された時間でタイマが再起動します。

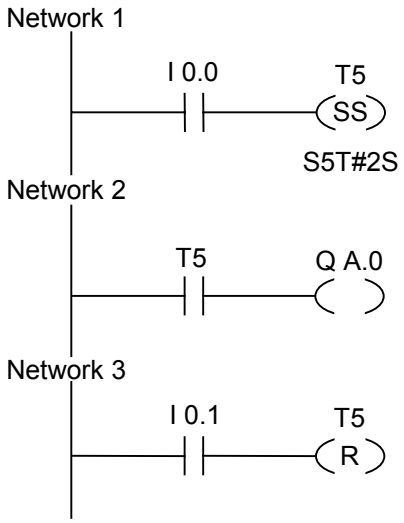
関連項目: "メモリ内のタイマの場所およびタイマのコンポーネント"および S_ODTS (拡張オンディレイ S5 タイマ)

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

13.11 ---(SS) 拡張オンディレイタイマコイル

例



入力 I0.0 の信号状態が"0"から"1"になると(RLO が信号立ち上がり)、タイマ T5 が起動します。タイマが終了する前に入力 I0.0 の信号状態が"0"から"1"になると、タイマは再トリガされます。タイマが終了すると、出力 Q4.0 は"1"になります。入力 I0.1 の信号状態が"1"になると、タイマ T5 はリセットされて停止し、残り時間は"0"にクリアされます。

13.12 ---(SF) オフディレイタイマコイル

シンボル

英語	ドイツ語
<T no.>	<T no.>
---(SF)	**---(SA)**
<time value>	<time value>

パラメータ	データタイプ	メモリ領域	説明
<T no.>	TIMER	T	タイマ識別番号。番号の範囲は CPU により異なる。
<time value>	S5TIME	I、Q、M、L、D	設定済みの時間

説明

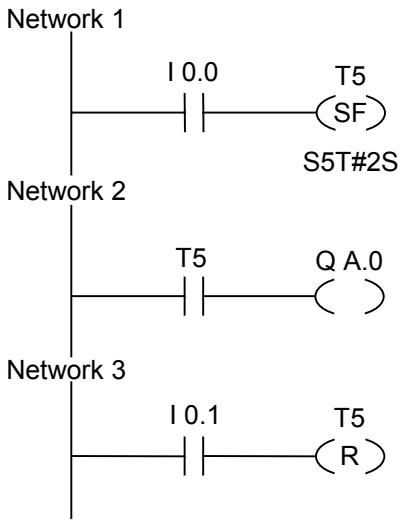
---(SF) (オフディレイタイマコイル)は、RLO が信号立ち下がりのときに、指定されたタイマを起動します。RLO が"1"のとき、または、**<time value>** の間隔内でタイマが実行しているとき、タイマは"1"になります。タイマの実行中に RLO が"0"から"1"になると、タイマはリセットされます。RLO が"1"から"0"に変わると、タイマは再起動します。

関連項目:「メモリ内のタイマの場所およびタイマのコンポーネント」および「S_OFFDT (オフディレイ S5 タイマ)」

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	-	-	-	-	-	0	-	-	0

例



13.12 ---(SF) オフディレイタイマコイル

入力 I0.0 の信号状態が"1"から"0"になると、タイマが起動します。

入力 I0.0 が"1"のとき、またはタイマの実行中、出力 Q4.0 の信号状態は"1"になります。入力 I0.1 の信号状態が"0"から"1"になると、タイマ T5 はリセットされます。タイマは停止し、タイマ値の残りの部分は"0"にクリアされます。

14 ワード論理命令

14.1 ワード論理命令の概要

説明

ワード論理演算命令により、ワード(16 ビット) やダブルワード(32 ビット) のペアをブールロジックに従ってビットごとに比較します。

出力 OUT の結果が 0 以外の場合、ステータスワードのビット CC1 は"1"に設定されます。

出力 OUT の結果が 0 の場合、ステータスワードのビット CC1 は"0"に設定されます。

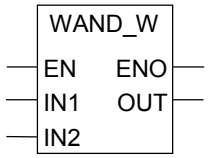
使用可能なワード論理命令を次に示します。

- WAND_W (ワード) AND ワード
- WOR_W (ワード) OR ワード
- WXOR_W (ワード) 排他的 OR ワード

- WAND_DW (ワード) AND ダブルワード
- WOR_DW (ワード) OR ダブルワード
- WXOR_DW (ワード)排他的 OR ダブルワード

14.2 WAND_W (ワード) AND ワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	WORD	I、Q、M、L、D	論理演算の最初の値
IN2	WORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	WORD	I、Q、M、L、D	論理演算から得られたワード

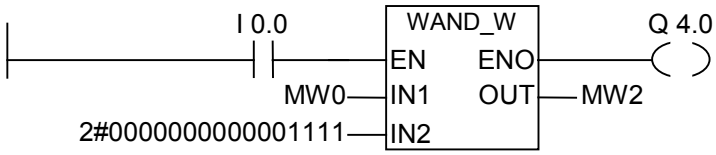
説明

WAND_W (AND ワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の論理積をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



命令は、I0.0 が"1"のときに実行されます。MW0 のビット 0～3 のみが対象となり、残りのビットは、IN2 ワードビットパターンによりマスクされます。

MW0 = 01010101 01010101

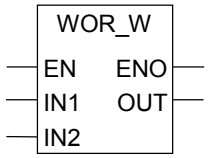
IN2 = 00000000 00001111

MW0 AND IN2 = MW2 = 00000000 00000101

命令が実行されると、Q4.0 は"1"になります。

14.3 WOR_W (ワード) OR ワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	WORD	I、Q、M、L、D	論理演算の最初の値
IN2	WORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	WORD	I、Q、M、L、D	論理演算から得られたワード

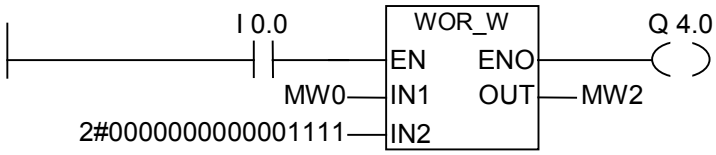
説明

WOR_W (OR ワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の論理和をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



命令は、I0.0 が"1"のときに実行されます。MW3 のビット 0～3 は"1"に設定され、残りのビットは変わりません。

MW0 = 01010101 01010101

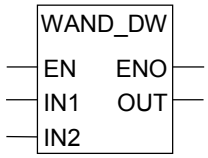
IN2 = 00000000 00001111

MW0 OR IN2=MW2 = 01010101 01011111

命令が実行されると、Q4.0 は"1"になります。

14.4 WAND_DW (ワード) AND ダブルワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DWORD	I、Q、M、L、D	論理演算の最初の値
IN2	DWORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	DWORD	I、Q、M、L、D	論理演算から得られたダブルワード

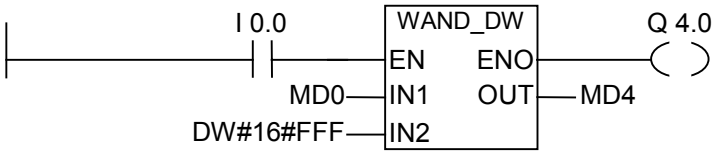
説明

WAND_DW (AND ダブルワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の論理積をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



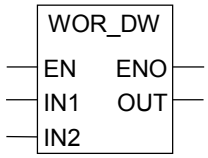
命令は、I0.0 が"1"のときに実行されます。MD0 のビット 0～11 のみが対象となり、残りのビットは、IN2 ビットパターンによりマスクされます。

MD0	=	01010101 01010101 01010101 01010101
IN2	=	00000000 00000000 00001111 11111111
MD0 AND IN2 = MD4	=	00000000 00000000 00000101 01010101

命令が実行されると、Q4.0 は"1"になります。

14.5 WOR_DW (ワード) OR ダブルワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DWORD	I、Q、M、L、D	論理演算の最初の値
IN2	DWORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	DWORD	I、Q、M、L、D	論理演算から得られたダブルワード

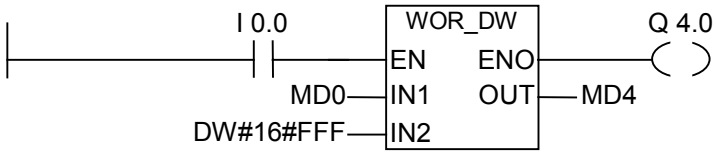
説明

WOR_DW (OR ダブルワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の論理和をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



命令は、I0.0 が"1"のときに実行されます。MD0 のビット 0～11 は"1"に設定され、残りのビットは変わりません。

MD0 = 01010101 01010101 01010101 01010101

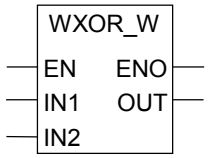
IN2 = 00000000 00000000 00001111 11111111

MD0 OR IN2 = MD4 = 01010101 01010101 01011111 11111111

命令が実行されると、Q4.0 は"1"になります。

14.6 WXOR_W (ワード) 排他的 OR ワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	WORD	I、Q、M、L、D	論理演算の最初の値
IN2	WORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	WORD	I、Q、M、L、D	論理演算から得られたワード

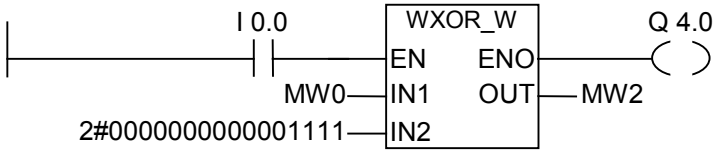
説明

WXOR_W (排他的 OR ワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の排他的論理和をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



I0.0 が"1"になると、命令が実行されます。

MW0 = 01010101 01010101

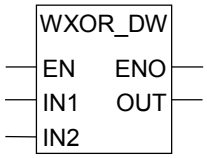
IN2 = 00000000 00001111

MW0 XOR IN2 = MW2 = 01010101 01011010

命令が実行されると、Q4.0 は"1"になります。

14.7 WXOR_DW (ワード) 排他的 OR ダブルワード

シンボル



パラメータ	データタイプ	メモリ領域	説明
EN	BOOL	I、Q、M、L、D	イネーブル入力
ENO	BOOL	I、Q、M、L、D	イネーブル出力
IN1	DWORD	I、Q、M、L、D	論理演算の最初の値
IN2	DWORD	I、Q、M、L、D	論理演算の 2 番目の値
OUT	DWORD	I、Q、M、L、D	論理演算から得られたダブルワード

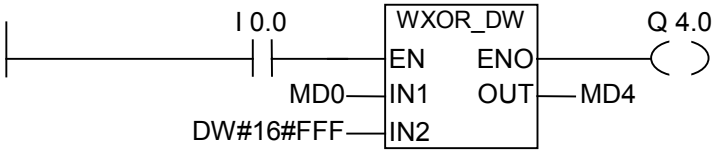
説明

WXOR_DW (排他的 OR ダブルワード)は、入力の有効化(EN)の信号状態が"1"のときに起動し、IN1 と IN2 のワード値の排他的論理和をビットごとに計算します。これらのワード値は、完全なビットパターンとして扱われます。計算結果は、出力 OUT で確認できます。ENO は、EN と同じロジック状態になります。

ステータスワード

	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	/FC
書き込みの内容:	1	x	0	0	-	x	1	1	1

例



I0.0 が"1"になると、命令が実行されます。

MD0 = 01010101 01010101 01010101 01010101

IN2 = 00000000 00000000 00001111 11111111

MW2 = MD0 XOR IN2 = 01010101 01010101 01011010 10101010

命令が実行されると、Q4.0 は"1"になります。

A 全 LAD 命令の概要

A.1 英語のプログラム表記法(インターナショナル)に従ってソートされた LAD 命令

英語のプログラム表記法	ドイツ語のプログラム表記法	プログラムエレメントカタログ	説明
---| |---	---| |---	ビットロジック命令	a 接点(アドレス)
---|/|---	---|/|---	ビットロジック命令	b 接点(アドレス)
---( )	---( )	ビットロジック命令	出力コイル
---(#)---	---(#)---	ビットロジック命令	中間出力
==0 ---| |---	==0 ---| |---	ステータスビット	リザルトビット=0
>0 ---| |---	>0 ---| |---	ステータスビット	リザルトビット>0
>=0 ---| |---	>=0 ---| |---	ステータスビット	リザルトビット>=0
<=0 ---| |---	<=0 ---| |---	ステータスビット	リザルトビット<=0
<0 ---| |---	<0 ---| |---	ステータスビット	リザルトビット<0
<>0 ---| |---	<>0 ---| |---	ステータスビット	リザルトビット<>0
ABS	ABS	浮動小数点命令	浮動小数点数の絶対値を求める
ACOS	ACOS	浮動小数点命令	アークコサイン値を求める
ADD_DI	ADD_DI	整数演算命令	倍長整数の加算
ADD_I	ADD_I	整数演算命令	整数の加算
ADD_R	ADD_R	浮動小数点命令	実数の加算
ASIN	ASIN	浮動小数点命令	アークサイン値を求める
ATAN	ATAN	浮動小数点命令	アークタンジェント値を求める
BCD_DI	BCD_DI	変換	BCD から倍長整数
BCD_I	BCD_I	変換	BCD から整数
BR ---| |---	BIE ---| |---	ステータスビット	BR ビット
----(CALL)	----(CALL)	プログラムコントロール	FC SFC のコイルからの呼び出し(パラメータなし)
CALL_FB	CALL_FB	プログラムコントロール	FB のボックスからの呼び出し
CALL_FC	CALL_FC	プログラムコントロール	FC のボックスからの呼び出し
CALL_SFB	CALL_SFB	プログラムコントロール	SFB のボックスからの呼び出し
CALL_SFC	CALL_SFC	プログラムコントロール	SFC のボックスからの呼び出し
----(CD)	----(ZR)	カウンタ	カウントダウンコイル
CEIL	CEIL	変換	切り上げ
CMP>=D	CMP>=D	比較	倍長整数の比較(==, <>, >, <, >=, <=)

英語のプログラム表記法	ドイツ語のプログラム表記法	プログラムエレメントカタログ	説明
CMP>=I	CMP>=I	比較	整数の比較(==, <>, >, <, >=, <=)
CMP >=R	CMP >=R	比較	実数の比較(==, <>, >, <, >=, <=)
COS	COS	浮動小数点命令	コサイン値を求める
----(CU)	---(ZV)	カウンタ	カウントアップコイル(CU)
DI_BCD	DI_BCD	変換	倍長整数から BCD
DI_R	DI_R	変換	倍長整数から浮動小数点へ
DIV_DI	DIV_DI	整数演算命令	倍長整数の除算
DIV_I	DIV_I	整数演算命令	整数の除算
DIV_R	DIV_R	浮動小数点命令	実数の除算
EXP	EXP	浮動小数点命令	指数値を求める
FLOOR	FLOOR	変換	切り下げ
I_BCD	I_BCD	変換	整数から BCD への変換
I_DI	I_DI	変換	整数から倍長整数への変換
INV_I	INV_I	変換	整数の 1 の補数
INV_DI	INV_DI	変換	倍長整数のビット反転
---(JMP)	---(JMP)	ジャンプ	条件なしジャンプ
---(JMP)	---(JMP)	ジャンプ	条件付きジャンプ
---(JMPN)	---(JMPN)	ジャンプ	ジャンプイフノット
LABEL	LABEL	ジャンプ	Label
LN	LN	浮動小数点命令	自然対数を求める
---(MCR>)	---(MCR>)	プログラムコントロール	マスタコントロールリレーのオフ
---(MCR<)	---(MCR<)	プログラムコントロール	マスタコントロールリレーのオン
---(MCRA)	---(MCRA)	プログラムコントロール	マスタコントロールリレーの開始
---(MCRD)	---(MCRD)	プログラムコントロール	マスタコントロールリレーの終了
MOD_DI	MOD_DI	整数演算命令	倍長整数の商余
MOVE	MOVE	移動	値の割り付け
MUL_DI	MUL_DI	整数演算命令	倍長整数の乗算
MUL_I	MUL_I	整数演算命令	整数の乗算
MUL_R	MUL_R	浮動小数点命令	実数の乗算
---(N)---	---(N)---	ビットロジック命令	立ち下がりパルス
NEG	NEG	ビットロジック命令	アドレス立ち下がりパルス
NEG_DI	NEG_DI	変換	倍長整数の符号反転(NEG_DI)
NEG_I	NEG_I	変換	整数の符号反転
NEG_R	NEG_R	変換	負の浮動小数点数
---| NOT |---	---| NOT |---	ビットロジック命令	パワーフローの反転
---(OPN)	---(OPN)	DB 呼び出し	データブロックを開く: DB または DI
OS ---| |---	OS ---| |---	ステータスビット	OS メモリビット
OV ---| |---	OV ---| |---	ステータスビット	OV ビット

英語のプログラム表記法	ドイツ語のプログラム表記法	プログラムエレメントカタログ	説明
---(P)---	---(P)---	ビットロジック命令	立ち上がり RLO 信号検出
POS	POS	ビットロジック命令	アドレス立ち上がりパルス
---(R)	---(R)	ビットロジック命令	リセットコイル
---(RET)	---(RET)	プログラムコントロール	リターン
ROL_DW	ROL_DW	シフト/ 循環	ダブルワード左回転
ROR_DW	ROR_DW	シフト/ 循環	ダブルワード右回転
ROUND	ROUND	変換	倍長整数の丸め
RS	RS	ビットロジック命令	リセット-セットフリップフロップ
---(S)	---(S)	ビットロジック命令	セットコイル
---(SAVE)	---(SAVE)	ビットロジック命令	RLO を BR メモリに保存
---(SC)	---(SZ)	カウンタ	カウンタ値の設定
S_CD	Z_RUECK	カウンタ	カウントダウン
S_CU	Z_VORW	カウンタ	カウントアップ
S_CUD	ZAEHLER	カウンタ	カウントアップダウン
---(SD)	---(SE)	タイマ	オンディレイタイマコイル
---(SE)	---(SV)	タイマ	拡張パルスタイマコイル
---(SF)	---(SA)	タイマ	オフディレイタイマコイル
SHL_DW	SHL_DW	シフト/ 循環	ダブルワード左シフト
SHL_W	SHL_W	シフト/ 循環	ワード左シフト
SHR_DI	SHR_DI	シフト/ 循環	倍長整数右シフト
SHR_DW	SHR_DW	シフト/ 循環	ダブルワード右シフト
SHR_I	SHR_I	シフト/ 循環	整数右シフト
SHR_W	SHR_W	シフト/ 循環	ワード右シフト
SIN	SIN	浮動小数点命令	サイン値を求める
S_ODT	S_EVERZ	タイマ	オンディレイ S5 タイマ
S_ODTS	S_SEVERZ	タイマ	拡張オンディレイ S5 タイマ
S_OFFDT	S_AVERZ	タイマ	オフディレイ S5 タイマ
---(SP)	---(SI)	タイマ	パルスタイマコイル
S_PEXT	S_VIMP	タイマ	拡張パルス S5 タイマ
S_PULSE	S_IMPULS	タイマ	パルス S5 タイマ
SQR	SQR	浮動小数点命令	平方を求める
SQRT	SQRT	浮動小数点命令	平方根を求める
SR	SR	ビットロジック命令	セット-リセットフリップフロップ
---(SS)	---(SS)	タイマ	保持型オンディレイタイマコイル
SUB_DI	SUB_DI	整数演算命令	倍長整数の減算
SUB_I	SUB_I	整数演算命令	整数の減算
SUB_R	SUB_R	浮動小数点命令	実数の減算
TAN	TAN	浮動小数点命令	タンジェント値を求める
TRUNC	TRUNC	変換	実数の倍長整数変換(小数部切り捨て)
UO ---| |---	UO ---| |---	ステータスビット	UO ビット

英語のプログラム表記法	ドイツ語のプログラム表記法	プログラムエレメントカタログ	説明
WAND_DW	WAND_DW	ワード論理命令	AND ダブルワード
WAND_W	WAND_W	ワード論理命令	AND ワード
WOR_DW	WOR_DW	ワード論理命令	OR ダブルワード
WOR_W	WOR_W	ワード論理命令	OR ワード
WXOR_DW	WXOR_DW	ワード論理命令	排他的 OR ダブルワード
WXOR_W	WXOR_W	ワード論理命令	排他的 OR ワード

A.2 ドイツ語のプログラム表記法(SIMATIC)に従ってソートされた LAD 命令

ドイツ語のプログラム表記法	英語のプログラム表記法	プログラムエレメントカタログ	説明
---| |---	---| |---	ビットロジック命令	a 接点(アドレス)
---|/|---	---|/|---	ビットロジック命令	b 接点(アドレス)
---( )	---( )	ビットロジック命令	出力コイル
---(#)---	---(#)---	ビットロジック命令	中間出力
==0 ---| |---	==0 ---| |---	ステータスビット	リザルトビット=0
>0 ---| |---	>0 ---| |---	ステータスビット	リザルトビット>0
>=0 ---| |---	>=0 ---| |---	ステータスビット	リザルトビット>=0
<=0 ---| |---	<=0 ---| |---	ステータスビット	リザルトビット<=0
<0 ---| |---	<0 ---| |---	ステータスビット	リザルトビット<0
<>0 ---| |---	<>0 ---| |---	ステータスビット	リザルトビット<>0
ABS	ABS	浮動小数点命令	浮動小数点数の絶対値を求める
ACOS	ACOS	浮動小数点命令	アークコサイン値を求める
ADD_DI	ADD_DI	整数演算命令	倍長整数の加算
ADD_I	ADD_I	整数演算命令	整数の加算
ADD_R	ADD_R	浮動小数点命令	実数の加算
ASIN	ASIN	浮動小数点命令	アークサイン値を求める
ATAN	ATAN	浮動小数点命令	アークタンジェント値を求める
BCD_DI	BCD_DI	変換	BCD から倍長整数
BCD_I	BCD_I	変換	BCD から整数
BIE ---| |---	BR ---| |---	ステータスビット	BR ビット
----(CALL)	----(CALL)	プログラムコントロール	FC SFC のコイルからの呼び出し(パラメータなし)
CALL_FB	CALL_FB	プログラムコントロール	FB のボックスからの呼び出し
CALL_FC	CALL_FC	プログラムコントロール	FC のボックスからの呼び出し
CALL_SFB	CALL_SFB	プログラムコントロール	SFB のボックスからの呼び出し
CALL_SFC	CALL_SFC	プログラムコントロール	SFC のボックスからの呼び出し
CEIL	CEIL	変換	切り上げ
CMP>=D	CMP>=D	比較	倍長整数の比較(==, <>, >, <, >=, <=)
CMP>=I	CMP>=I	比較	整数の比較(==, <>, >, <, >=, <=)
CMP >=R	CMP >=R	比較	実数の比較(==, <>, >, <, >=, <=)
COS	COS	浮動小数点命令	コサイン値を求める
DI_BCD	DI_BCD	変換	倍長整数から BCD

ドイツ語のプログラム表記法	英語のプログラム表記法	プログラムエレメントカタログ	説明
DI_R	DI_R	変換	倍長整数から浮動小数点へ
DIV_DI	DIV_DI	整数演算命令	倍長整数の除算
DIV_I	DIV_I	整数演算命令	整数の除算
DIV_R	DIV_R	浮動小数点命令	実数の除算
EXP	EXP	浮動小数点命令	指数値を求める
FLOOR	FLOOR	変換	切り下げ
I_BCD	I_BCD	変換	整数から BCD への変換
I_DI	I_DI	変換	整数から倍長整数への変換
INV_I	INV_I	変換	整数の 1 の補数
INV_DI	INV_DI	変換	倍長整数のビット反転
---(JMP)	---(JMP)	ジャンプ	条件付きジャンプ
---(JMP)	---(JMP)	ジャンプ	条件なしジャンプ
---(JMPN)	---(JMPN)	ジャンプ	ジャンプイフノット
LABEL	LABEL	ジャンプ	Label
LN	LN	浮動小数点命令	自然対数を求める
---(MCR>)	---(MCR>)	プログラムコントロール	マスタコントロールリレーのオフ
---(MCR<)	---(MCR<)	プログラムコントロール	マスタコントロールリレーのオン
---(MCRA)	---(MCRA)	プログラムコントロール	マスタコントロールリレーの開始
---(MCRD)	---(MCRD)	プログラムコントロール	マスタコントロールリレーの終了
MOD_DI	MOD_DI	整数演算命令	倍長整数の商余
MOVE	MOVE	移動	値の割り付け
MUL_DI	MUL_DI	整数演算命令	倍長整数の乗算
MUL_I	MUL_I	整数演算命令	整数の乗算
MUL_R	MUL_R	浮動小数点命令	実数の乗算
---(N)---	---(N)---	ビットロジック命令	立ち下がりパルス
NEG	NEG	ビットロジック命令	アドレス立ち下がりパルス
NEG_DI	NEG_DI	変換	倍長整数の符号反転(NEG_DI)
NEG_I	NEG_I	変換	整数の符号反転
NEG_R	NEG_R	変換	負の浮動小数点数
---| NOT |---	---| NOT |---	ビットロジック命令	パワーフローの反転
---(OPN)	---(OPN)	DB 呼び出し	データブロックを開く: DB または DI
OS ---| |---	OS ---| |---	ステータスビット	OS メモリビット
OV ---| |---	OV ---| |---	ステータスビット	OV ビット
---(P)---	---(P)---	ビット論理命令	立ち上がり RLO 信号検出
POS	POS	ビットロジック命令	アドレス立ち上がりパルス

ドイツ語のプログラム表記法	英語のプログラム表記法	プログラムエレメントカタログ	説明
---(R)	---(R)	ビットロジック命令	リセットコイル
---(RET)	---(RET)	プログラムコントロール	リターン
ROL_DW	ROL_DW	シフト/ 循環	ダブルワード左回転
ROR_DW	ROR_DW	シフト/ 循環	ダブルワード右回転
ROUND	ROUND	変換	倍長整数の丸め
RS	RS	ビットロジック命令	リセット-セットフリップフロップ
---(S)	---(S)	ビットロジック命令	セットコイル
---(SA)	---(SF)	タイマ	オフディレイタイマコイル
---(SAVE)	---(SAVE)	ビットロジック命令	RLO を BR メモリに保存
S_AVERZ	S_OFFDT	タイマ	オフディレイ S5 タイマ
---(SE)	---(SD)	タイマ	オンディレイタイマコイル
S_EVERZ	S_ODT	タイマ	オンディレイ S5 タイマ
SHL_DW	SHL_DW	シフト/ 循環	ダブルワード左シフト
SHL_W	SHL_W	シフト/ 循環	ワード左シフト
SHR_DI	SHR_DI	シフト/ 循環	倍長整数右シフト
SHR_DW	SHR_DW	シフト/ 循環	ダブルワード右シフト
SHR_I	SHR_I	シフト/ 循環	整数右シフト
SHR_W	SHR_W	シフト/ 循環	ワード右シフト
---(SI)	---(SP)	タイマ	パルスタイマコイル
S_IMPULS	S_PULSE	タイマ	パルス S5 タイマ
SIN	SIN	浮動小数点命令	サイン値を求める
SQR	SQR	浮動小数点命令	平方を求める
SQRT	SQRT	浮動小数点命令	平方根を求める
SR	SR	ビットロジック命令	セット-リセットフリップフロップ
---(SS)	---(SS)	タイマ	保持型オンディレイタイマコイル
S_SEVERZ	S_ODTS	タイマ	拡張オンディレイ S5 タイマ
SUB_DI	SUB_DI	整数演算命令	倍長整数の減算
SUB_I	SUB_I	整数演算命令	整数の減算
SUB_R	SUB_R	浮動小数点命令	実数の減算
---(SV)	---(SE)	タイマ	拡張パルスタイマコイル
S_VIMP	S_PEXT	タイマ	拡張パルス S5 タイマ
---(SZ)	---(SC)	カウンタ	カウンタ値の設定
TAN	TAN	浮動小数点命令	タンジェント値を求める
TRUNC	TRUNC	変換	実数の倍長整数変換(小数部切り捨て)
UO ---| |---	UO ---| |---	ステータスビット	UO ビット
WAND_DW	WAND_DW	ワード論理命令	AND ダブルワード
WAND_W	WAND_W	ワード論理命令	AND ワード
WOR_DW	WOR_DW	ワード論理命令	OR ダブルワード

ドイツ語のプログラム表記法	英語のプログラム表記法	プログラムエレメントカタログ	説明
WOR_W	WOR_W	ワード論理命令	OR ワード
WXOR_DW	WXOR_DW	ワード論理命令	排他的 OR ダブルワード
WXOR_W	WXOR_W	ワード論理命令	排他的 OR ワード
ZAEHLER	S_CUD	カウンタ	カウントアップダウン
----(ZR)	----(CD)	カウンタ	カウントダウンコイル
Z_RUECK	S_CD	カウンタ	カウントダウン
---(ZV)	----(CU)	カウンタ	カウントアップコイル(CU)
Z_VORW	S_CU	カウンタ	カウントアップ

B プログラミング例

B.1 プログラミングの概要例

実際の応用例

このマニュアルで説明するラダーロジック命令は、それぞれ特定の操作をトリガします。これらの命令を組み合わせて１つのプログラムにすると、各種のオートメーションタスクを実行できます。この章では、ラダーロジック命令を使った以下のような実践的な応用例を説明します。

- ビット論理命令によるコンベアベルトのコントロール
- ビットロジック命令を使用した搬送機ベルトの走行方向の検知
- タイマ命令を使用したクロックパルスの生成
- カウンタ命令と比較命令を使用した格納庫の管理
- 整数演算命令による問題の解決
- オーブンの加熱時間の設定

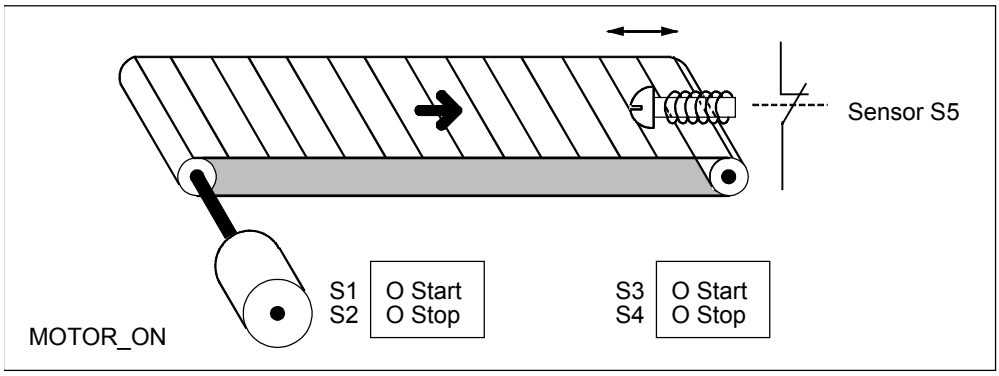
使用する命令

ニーモニック	プログラムエレメントカタログ	説明
WAND_W	ワード論理命令	ワードの論理積
WOR_W	ワード論理命令	ワードの論理和
---(CD)	カウンタ	カウントダウンコイル
---(CU)	カウンタ	カウントアップコイル(CU)
---(R)	ビット論理命令	リセットコイル
---(S)	ビット論理命令	セットコイル
---(P)	ビット論理命令	立ち上がり RLO 信号検出
ADD_I	浮動小数点命令	整数の加算
DIV_I	浮動小数点命令	整数の除算
MUL_I	浮動小数点命令	整数の乗算
CMP <=I、CMP >=I	比較	整数の比較
---| |---	ビット論理命令	a 接点
---| / |---	ビット論理命令	b 接点
—( )	ビット論理命令	出力コイル
---(JMPN)	ジャンプ	ジャンプイフノット
---(RET)	プログラムコントロール	リターン
MOVE	移動	値の割り付け
--- (SE)	タイマ	拡張パルスタイマコイル

B.2 例: ビットロジック命令

例 1: 搬送機ベルトの制御

以下の図に、電動式搬送機ベルトを示します。ベルトの始点には、START 用 S1 ボタンと STOP 用 S2 ボタンの 2 つの押しボタンがあります。ベルトの終点には、START 用 S3 スイッチと STOP 用 S4 スイッチの 2 つの押しボタンスイッチもあります。どちらの端からでもベルトを起動または停止できます。また、ベルト上の物体が終点に達すると、センサ S5 がこれを感知し、ベルトを停止させます。



絶対プログラミングとシンボルプログラミング

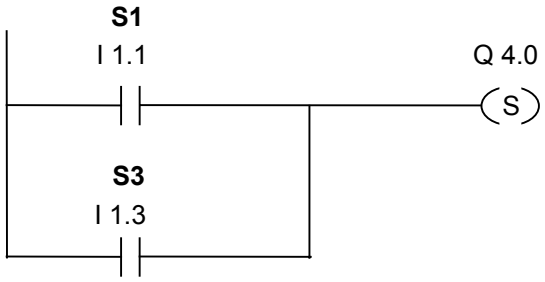
コンベアベルトをコントロールするプログラムを記述するには、コンベアシステムの各種コンポーネントを表す**絶対値**または**シンボル**を使用します。

シンボルテーブルを 1 つ作成して、選択したシンボルと絶対値を対応させる必要があります。STEP 7 オンラインヘルプを参照してください。

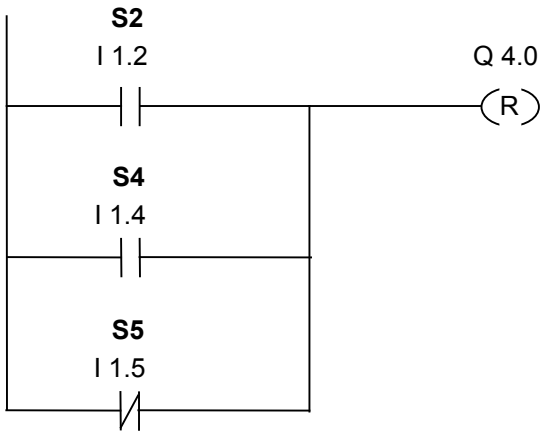
システムコンポーネント	絶対アドレス	シンボル	シンボルテーブル
START ボタン	I 1.1	S1	I 1.1 S1
STOP ボタン	I 1.2	S2	I 1.2 S2
START ボタン	I 1.3	S3	I 1.3 S3
STOP ボタン	I 1.4	S4	I 1.4 S4
センサ	I 1.5	S5	I 1.5 S5
Motor	Q 4.0	MOTOR_ON	Q 4.0 MOTOR_ON

コンベアベルトをコントロールするためのラダーロジックプログラム

ネットワーク 1: いずれかの開始スイッチを押すと、モータがオンになります。

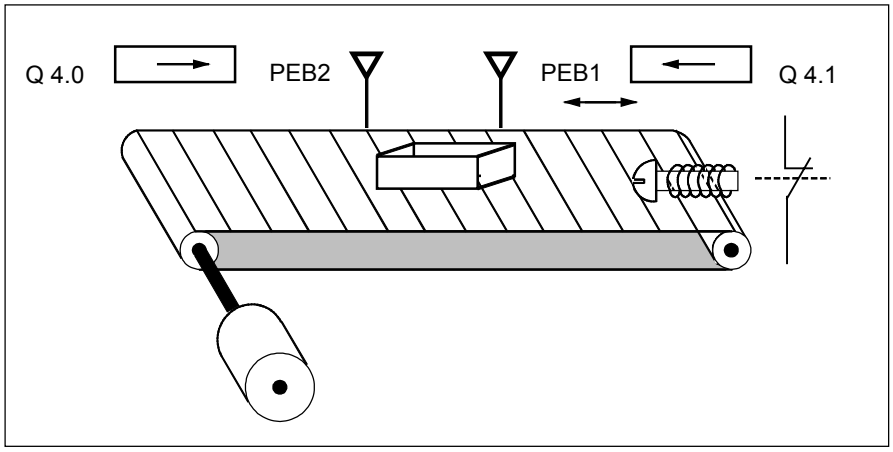


ネットワーク 2: いずれかの停止スイッチを押すか、ベルトの終端にある b 接点を開くと、モータがオフになります。



例 2: コンベアベルトの方向の検出

以下の図に、光電スイッチを 2 個(PEB1 と PEB2)を装備した搬送機ベルトを示します。この 2 個の光電スイッチは、ベルト上のパッケージの移動方向を検知できます。各光電スイッチは、a 接点と同じように機能します。



絶対プログラミングとシンボルプログラミング

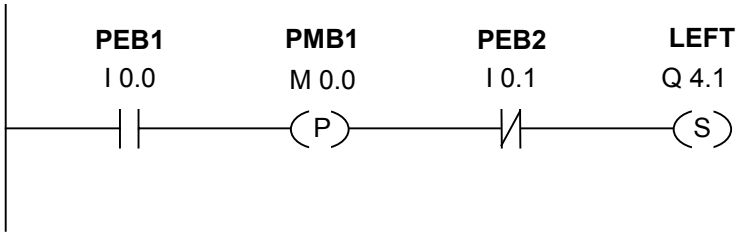
コンベアベルトシステムの方向表示を有効にするプログラムを記述するには、コンベアシステムの各種コンポーネントを表す**絶対値**または**シンボル**を使用します。

シンボルテーブルを 1 つ作成して、選択したシンボルと絶対値を対応させる必要があります。STEP 7 オンラインヘルプを参照してください。

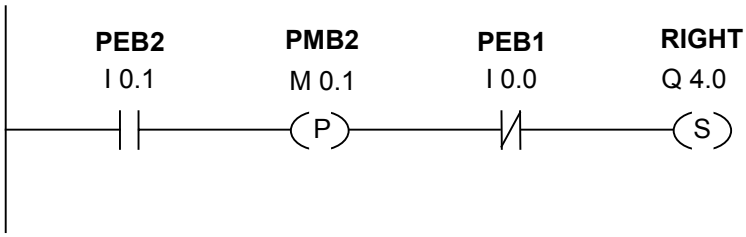
システムコンポーネント	絶対アドレス	シンボル	シンボルテーブル
光電スイッチ 1	I 0.0	PEB1	I 0.0 PEB1
光電スイッチ 2	I 0.1	PEB2	I 0.1 PEB2
右移動の表示	Q 4.0	RIGHT	Q 4.0 RIGHT
左移動の表示	Q 4.1	LEFT	Q 4.1 LEFT
パルスメモリビット 1	M 0.0	PMB1	M 0.0 PMB1
パルスメモリビット 2	M 0.1	PMB2	M 0.1 PMB2

コンベアベルトの方向を検出するためのラダーロジックプログラム

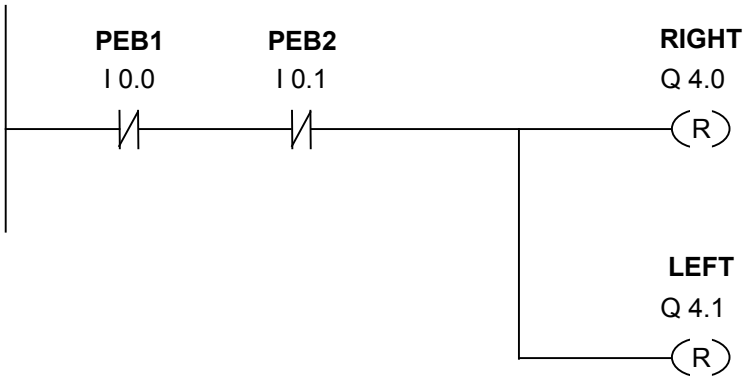
ネットワーク 1: 入力 I 0.0 の信号状態が 0 から 1(信号立ち上がり)に移行すると同時に、入力 I 0.1 の信号状態が 0 になっている場合は、ベルト上の荷物が左に移動しています。



ネットワーク 2: 入力 I 0.1 の信号状態が 0 から 1(信号立ち上がり)に移行すると同時に、入力 I 0.0 の信号状態が 0 になっている場合は、ベルト上の荷物が右に移動しています。物体が光電バリアの一方を通過するということは、2 個の光電バリアの間に物体があるということです。



ネットワーク 3: どちらの光電バリアも遮断されていない場合は、バリア間に荷物がないことになります。この場合、走行方向は表示されません。



B.3 例: タイマ命令

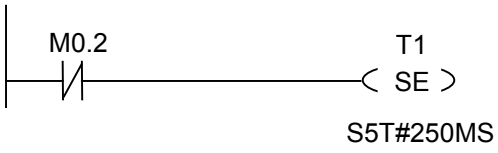
クロックパルス ジェネレータ

周期反復信号の生成が必要な場合、クロックパルスジェネレータまたはフラッシャ信号を使用できます。これは、表示ランプの点滅を制御する信号システムによく使用されています。

S7-300 を使用する場合､特殊なオーガニゼーションブロックでタイムドリブン処理を使用すれば、クロックパルスジェネレータファンクションを実行できます。ただし、次の FBD プログラムに示す例では、タイマファンクションを使用してクロックパルスを生成する方法を説明します。このサンプルプログラムで、タイマによるフリーホイーリングクロックパルスジェネレータの実現方法について説明します。

クロックパルスを生成するためのラダーロジックプログラム(パルス効率係数 1:1)

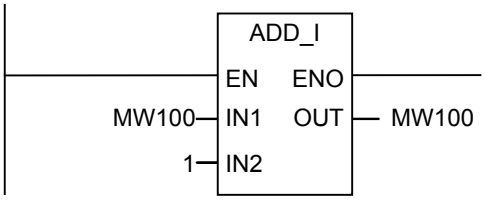
ネットワーク 1: タイマ T1 の信号状態が 0 の場合は、時間値 250 ms を T1 にロードし、T1 を拡張パルスタイマとして起動します。



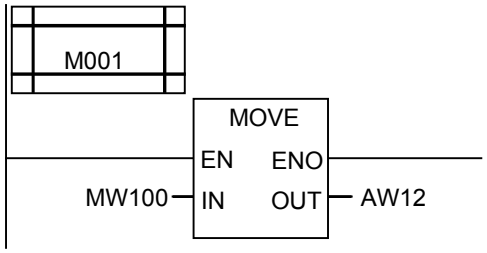
ネットワーク 2: タイマの状態は、補助メモリマーカーに一時的に保存されます。

ネットワーク 3: タイマ T1 の信号状態が 1 の場合は、ジャンプラベル M001 にジャンプします。

ネットワーク 4: タイマ T1 が切れると、メモリワード 100 が 1 増えます。

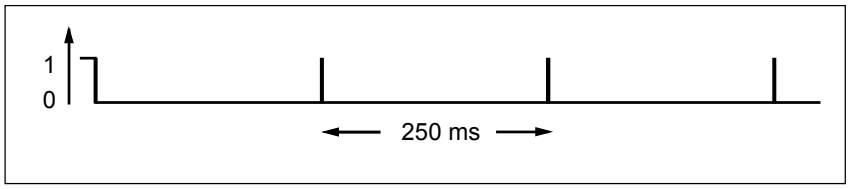


ネットワーク 5: **MOVE** 命令を使用すると、出力 Q12.0～Q13.7 で異なるクロック周波数を出力できます。



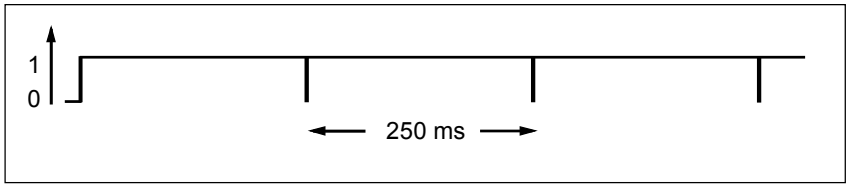
信号チェック

タイマ T1 の信号チェックを行うと、オープナーM0.2 に関して次の論理演算結果(RLO)が生成されます。



タイマは満了すると直ちに再起動します。このため、---| / |---M0.2 によって実行された信号チェックにより、信号状態が短時間だけ 1 になります。

否定(反転)RLO:



250 ms ごとに、RLO ビットが 0 になります。ジャンプは無視され、メモリワード MW100 の値が 1 増えます。

特定周波数への到達

メモリバイト MB101 および MB100 の各ビットから、次の周波数を実現することができます。

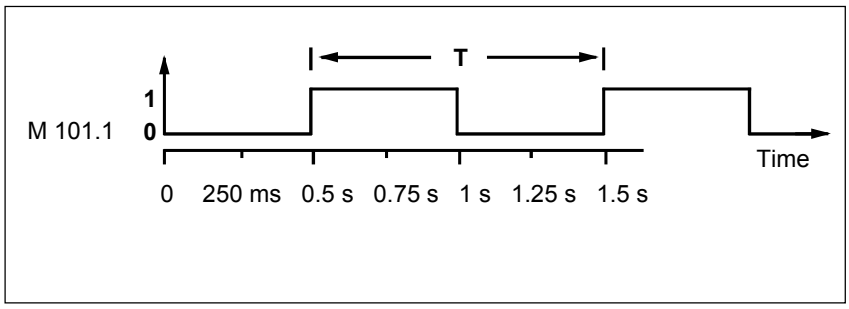
MB101/MB100 のビット	周波数(Hz)	パルス幅
M 101.0	2.0	0.5 s (250 ms オン/ 250 ms オフ)
M 101.1	1.0	1 s (0.5 s オン/0.5 s オフ)
M 101.2	0.5	2 s (1 s オン/ 1 s オフ)
M 101.3	0.25	4 s (2 s オン/ 2 s オフ)
M 101.4	0.125	8 s (4 s オン/ 4 s オフ)
M 101.5	0.0625	16 s (8 s オン/ 8 s オフ)
M 101.6	0.03125	32 s (16 s オン/ 16 s オフ)
M 101.7	0.015625	64 s (32 s オン/ 32 s オフ)
M 100.0	0.0078125	128 s (64 s オン/64 s オフ)
M 100.1	0.0039062	256 s (128 s オン/128 s オフ)
M 100.2	0.0019531	512 s (256 s オン/256 s オフ)
M 100.3	0.0009765	1024 s (512 s オン/512 s オフ)
M 100.4	0.0004882	2048 s (1024 s オン/1024 s オフ)
M 100.5	0.0002441	4096 s (2048 s オン/2048 s オフ)
M 100.6	0.000122	8192 s (4096 s オン/4096 s オフ)
M 100.7	0.000061	16384 s (8192 s オン/8192 s オフ)

メモリ MB101 の各ビットの信号状態

スキャンサイクル	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0	時間値 (単位: ms)
0	0	0	0	0	0	0	**0**	0	250
1	0	0	0	0	0	0	**0**	1	250
2	0	0	0	0	0	0	1	0	250
3	0	0	0	0	0	0	1	1	250
4	0	0	0	0	0	1	**0**	0	250
5	0	0	0	0	0	1	**0**	1	250
6	0	0	0	0	0	1	1	0	250
7	0	0	0	0	0	1	1	1	250
8	0	0	0	0	1	0	**0**	0	250
9	0	0	0	0	1	0	**0**	1	250
10	0	0	0	0	1	0	1	0	250
11	0	0	0	0	1	0	1	1	250
12	0	0	0	0	1	1	**0**	0	250

MB 101 (M 101.1)のビット 1 の信号状態

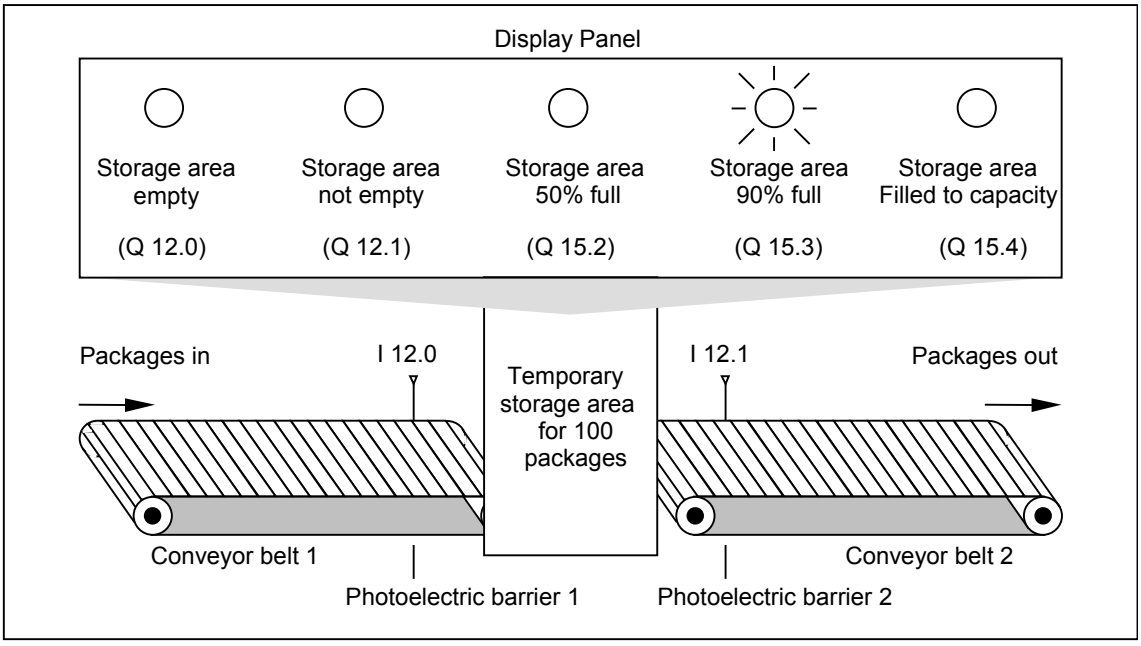
周波数= 1/T = 1/1 s = 1 Hz



B.4 例: カウンタ命令と比較命令

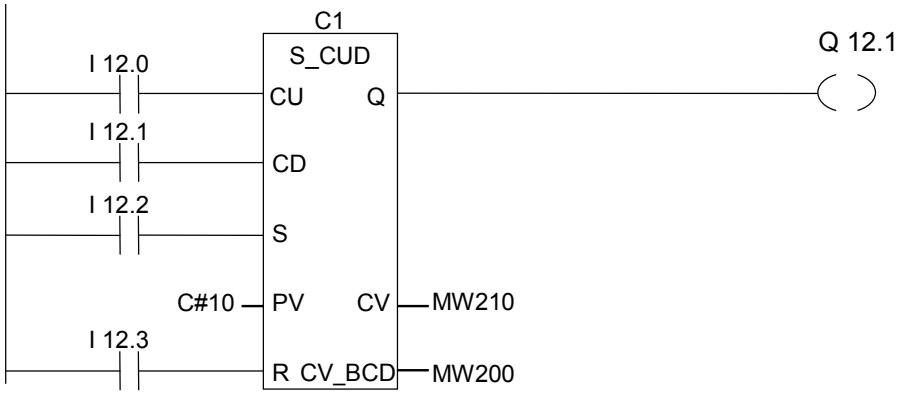
カウンタ命令と比較命令を使用した格納庫の管理

以下の図に、搬送機ベルトを 2 つ装備し、この 2 つのベルトの間にテンポラリ格納庫を装備したシステムを示します。搬送機ベルト 1 は、パッケージを格納庫まで配送します。搬送機ベルト 1 の末端、格納庫側にある光電スイッチにより、格納庫に搬入された荷物の数が確定されます。搬送機ベルト 2 は、パッケージをこのテンポラリ格納庫から積載ドックへ搬送します。パッケージは、この積載ドックからトラックで発送され、顧客に配送されます。搬送機ベルト 2 の末端、格納庫側にある光電スイッチにより、格納庫から積載ドックへ搬出された荷物の数が確定されます。テンポラリ格納庫の格納率は、表示パネルの 5 つのランプで示されます。



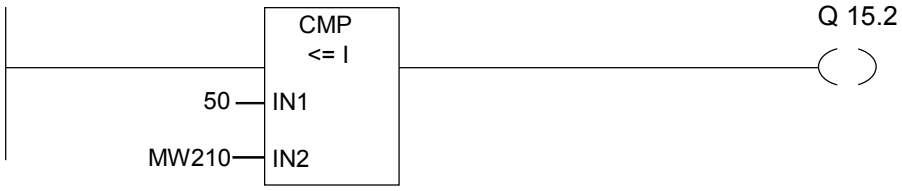
表示パネルの表示ランプを作動させるためのラダーロジックプログラム

ネットワーク 1: カウンタ C1 は、入力 CU の信号状態が“0”から“1”に変わるたびにカウントを増やし、入力 CD の信号状態が“0”から“1”に変わるたびにカウントを減らします。入力 S の信号状態が“0”から”1”に変わると、カウンタ値が値 PV に設定されます。入力 R の信号状態が"0"から"1"に変わると、カウンタ値が"0"にリセットされます。MW200 には、カウンタの C1 の現在値が出力されます。Q12.1 は、“格納エリアが空でない”ことを示します。

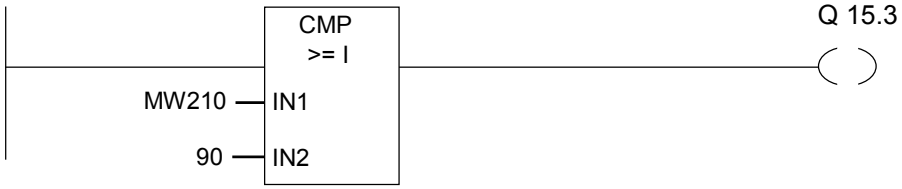


ネットワーク 2: Q12.0 は、"格納エリアが空"であることを示します。

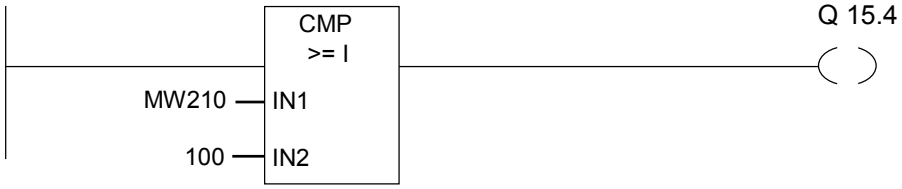
ネットワーク 3: 50 がカウンタ値よりも少ないかまたは等しい(つまり、現在のカウンタ値が 50 以上である)場合は、“格納エリア 50%充填”を示す表示ランプが点灯します。



ネットワーク 4: カウンタ値が 90 以上の場合は、“格納エリア 90%充填”を示す表示ランプが点灯します。



ネットワーク 5: カウンタ値が 100 以上の場合は、“格納エリア満杯” を示す表示ランプが点灯します。



B.5 例: 整数値演算命令

演算問題の解決

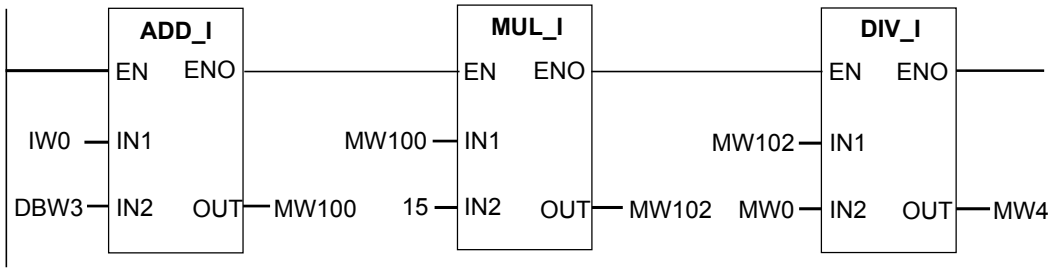
このサンプルプログラムでは、3 つの整数値演算命令を使用して、以下の方程式と同じ結果を求める方法を示します。

MW4 = ((IW0 + DBW3) x 15) / MW0

ラダーロジックプログラム

ネットワーク 1: データブロック DB1 を開きます。

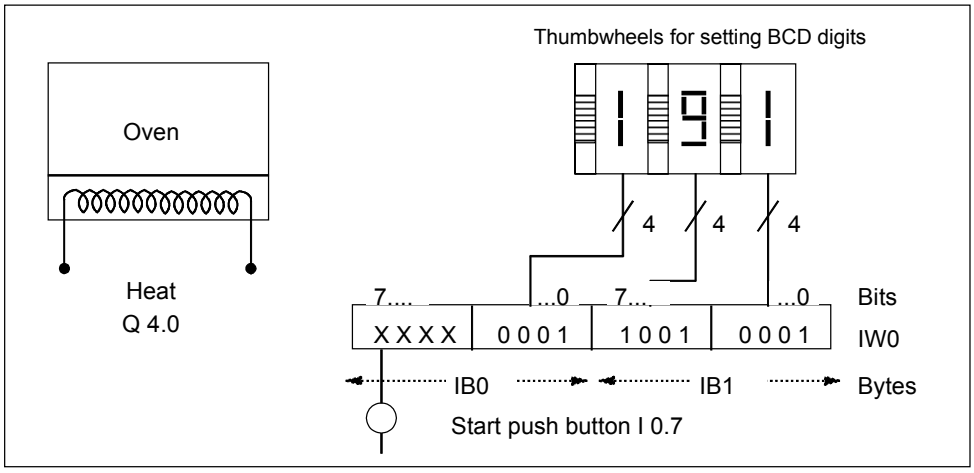
ネットワーク 2: 入力ワード IW0 が共有データワード DBW3 に加算され(データブロックが定義済みで開いていること)、合計がメモリワード MW100 にロードされます。次に、MW100 に 15 が掛けられて、その積がメモリダブルワード MW102 にセーブされます。MW102 は MW0 で割られて、その商は MW4 にセーブされます。



B.6 例: ワードロジック命令

オーブンの加熱

オペレータが START 押しボタンを押すと、オーブンは加熱を開始します。オペレータは、図に示されているサムホールスイッチを使用すれば、加熱時間を設定できます。設定した値は、2 進化 10 進数(BCD)フォーマットの秒数で示されます。



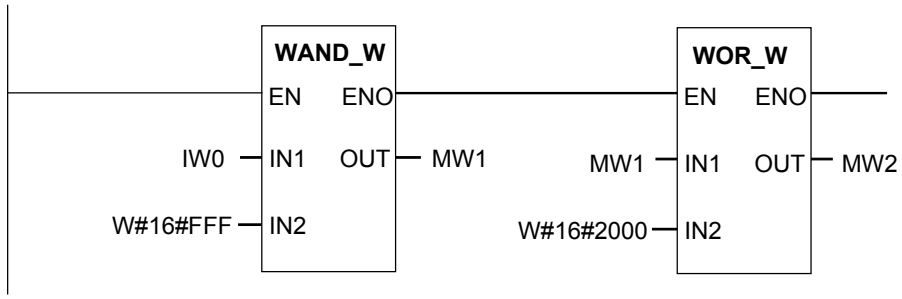
システムコンポーネント	絶対アドレス
START 押しボタン	I 0.7
1 の位のサムロータリースイッチ	I 1.0 ～ I 1.3
10 の位のサムホイール	I 1.4 ～ I 1.7
100 の位のサムロータリースイッチ	I 0.0 ～ I 0.3
加熱開始	Q 4.0

ラダーロジックプログラム

ネットワーク 1: タイマが作動している場合は、ヒーターがオンになります。

ネットワーク 2: タイマが作動している場合は、**Return** 命令によりここで処理が終了します。

ネットワーク 3: 入力ビット I 0.4～I 0.7 をマスキングします(つまり、これらを 0 にリセットします)。サムホイール入力のこれらのビットは使用されません。サムホイール入力の 16 ビットは、**WAND_W** 命令に従って W#16#0FFF と組み合わされます。この結果は、メモリワード MW1 に記録されます。秒単位のタイムベースと設定するために、設定値は **WOR_W** 命令に従って W#16#2000 と組み合わされ、ビット 13 が 1 に、ビット 12 が 0 に設定されます。



ネットワーク 4 : 開始押しボタンが押された場合は、タイマ T1 を拡張パルスタイマとして起動し、事前設定値として(上記の論理命令から導出された)メモリワード MW2 をロードします。

B.6 例: ワードロジック命令

C ラダーロジックでの作業

C.1 ブロックのタイプ

プログラムはブロックに分割されているため、プログラムの構造化が容易です。ブロックは、プログラムの独立したパートで、それぞれが特定の機能をもっています。STEP 7 では、信号処理の命令ステートメントをもつ論理ブロック(OB、FB、SFB、FC、SFC)とデータの保存に使用されるデータブロック(DB、DI)に分類しています。

論理ブロックは、プログラムやサブルーチンのソフトウェアモジュールとして使用されます。ロジックブロックの最上位レベルは、OB1(オーガニゼーションブロック 1)です。あるブロックが別のブロックから呼び出された場合、両方のブロックの変数宣言テーブルをプログラミングすることにより、ブロック間でパラメータを交換することができます。たとえば、呼び出されるブロックは、呼び出し側のブロックから様々な入力/出力パラメータを受け取るようにプログラムできます。呼び出されたときに入力値を処理して、その結果とプログラムコントロールを呼び出し側のブロックに返すことができます。

ステートメントの CALL FC1(ファンクション呼び出し)と CALL FB1、DB2(ファンクションブロック呼び出し)には、相違点があります。ファンクションは前回の処理を保存しないため、別のブロックが呼び出されると、処理中であってもローカルメモリ内の呼び出しパラメータとその割り付けが削除されます。一方、ファンクションブロックの呼び出しには、インスタンスデータブロックが関連付けられています。インスタンスデータブロックには、FB 変数宣言テーブルに従って、ブロック固有の変数が保存されます。

データブロックには次の 2 種類があります。

1. ユーザーデータが保存されているデータブロック。ユーザーデータは、ユーザー独自の方法で編成できます。このデータブロックは、論理ブロックで開き、オペレーションを読み書きできます。

2. 呼び出しパラメータとスタティックローカルデータの保存に使用するインスタンスデータブロック。このパラメータとデータは、ファンクションブロックの実行インスタンスで使用されます。

データブロックおよびロジックブロックは、STL エディタ、ラダーエディタ、または FBD エディタで作成します。STEP 7 には、様々な標準ブロックがあらかじめ組み込まれています。

C.2 EN/ENO 機構

FBD/LAD ボックスのイネーブル(EN)およびイネーブル出力(ENO)は、BR ビットによって取得されます。

EN と ENO が接続される場合、次のことが適用されます。

ENO = EN AND NOT (ボックスエラー)

エラーは発生しない(ボックスエラー = 0)の場合、ENO = EN。

EN/ENO メカニズムは以下のものに対して使用されます。

- 数値演算命令
- 転送命令および変換命令
- シフト命令および循環命令
- ブロックの呼び出し

このメカニズムは、以下のものに使用**されません**。

- 比較
- カウンタ
- タイマ

ボックスにおける実際の命令では、EN/ENO メカニズムに対して追加の STL 命令が生成されます。これは、既存の前の論理操作およびその後の論理操作に依存します。下記に、考えられる 4 つの加算機の使用例を示しています。

4. EN と ENO が接続されている加算器
5. EN を接続し ENO を接続しない加算機
6. EN を接続しないで ENO を接続した加算機
7. EN も ENO も接続されていない加算器

ブロックの作成に関する注記

FBD または LAD に呼び出すブロックをプログラムする場合、そのブロックを終了する時点で BR ビットを必ず設定する必要があります。4 番目の例は、自動的に実行される事例ではないことを示しています。BR は常に EN/ENO メカニズムによって上書きされるので、BR をメモリビットとして使用できません。代わりに、テンポラリ変数を使用します。エラーが発生すると必ず、このエラーをテンポラリ変数に保存します。この変数を 0 に初期化します。命令が失敗するとブロック全体のエラーとなるポイントがある場合、ブロック内のこうしたポイントごとに、EN/ENO によってこの変数を設定します。この目的には、NOT および SET コイルで十分です。このブロックプログラムが終わると、次のネットワークが処理されます。

```
end:   AN error
       SAVE
```

どの場合も必ずこのネットワークが処理されます。この結果、ブロック内では BEC を使用できないため、このネットワークをスキップします。

C.2.1 EN と ENO が接続されている加算器

加算機に EN と ENO が接続されている場合、次の STL 命令がトリガされます。

```
1    A   I    0.0  // EN 接続
2    JNB _001      // RLO を BR にシフトし、RLO = 0 でジャンプ
3    L     in1     // ボックスパラメータ
4    L     in2     // ボックスパラメータ
5    +I            // 実際の追加分
6    T     out     // ボックスパラメータ
7    AN    OV      // エラーの認識
8    SAVE          // エラーを BR に保存する
9    CLR           // ファーストチェック
10 _001: A     BR  // BR を RLO もシフト
11   =     Q   4.0
```

行 1 に続く RLO には、前の論理演算結果が含まれます。JNB 命令は、RLO を BR ビットにコピーして、最初のチェクビットを設定します。

- RLO = 0 の場合、プログラムは行 10 にジャンプし、A BR を再開します。加算は実行されません。行 10 で、BR が RLO に再度コピーされ、0 が出力に割り付けられます。
- RLO が 1 のとき、プログラムはジャンプしません。すなわち、加算が実行されます。行 7 で、プログラムは、加算中にエラーが発生したかどうか評価します。その後、行 8 で、この結果を BR に保存します。行 9 に、最初のチェックビットが設定されます。行 10 で、BR ビットが RLO にコピーされ、出力に加算演算が正常に終了したかどうか示されます。
  行 10 と 11 では BR ビットは変更されません。このため、加算が成功したかどうかも示されます。

C.2.2 EN を接続し ENO を接続しない加算機

加算機に EN は接続されているが ENO は接続されていない場合、次の STL 命令がトリガされます。

```
1    A   I   0.0   // EN 接続
2    JNB _001      // RLO を BR にシフトし、RLO = 0 でジャンプ
3    L     in1     // ボックスパラメータ
4    L     in2     // ボックスパラメータ
5    +I            // 実際の追加分
6    T     out     // ボックスパラメータ
7    _001: NOP   0
```

行 1 に続く RLO には、前の論理演算結果が含まれます。JNB 命令は、RLO を BR ビットにコピーして、最初のチェクビットを設定します。

- RLO が 0 のとき、プログラムは行 7 にジャンプするため、加算は実行されません。RLO と BR は 0 です。
- RLO が 1 であった場合、プログラムはジャンプしないので、加算が実行されることを意味します。このプログラムは加算演算中にエラーが発生したかどうかを評価しません。RLO および BR は 1 になります。

C.2.3 EN を接続しないで ENO を接続した加算機

加算機に EN を接続しないが ENO を接続した場合、以下の STL 命令がトリガされます。

```
1    L     in1     // ボックスパラメータ
2    L     in2     // ボックスパラメータ
3    +I            // 実際の追加分
4    T     out     // ボックスパラメータ
5    AN    OV      // エラーの認識
6    SAVE          // エラーを BR に保存する
7    CLR           // ファーストチェック
8    A     BR      // BR を RLO もシフト
9    = Q 4.0
```

すべての場合、加算演算が実行されます。行 5 で、プログラムは加算演算中にエラーが発生したかどうかを評価し、行 6 でその結果を BR に保存します。行 7 は最初のチェックビットを設定します。行 8 で、BR ビットが RLO にコピーされ、出力に加算演算が正常に終了したかどうか示されます。

行 8 と 9 では BR ビットは変更されません。このため、加算が成功したかどうかも示されます。

C.2.4 EN も ENO も接続されていない加算器

加算機に EN も ENO も接続されていない場合、次の STL 命令がトリガされます。

```
1    L     in1      // ボックスパラメータ
2    L     in2      // ボックスパラメータ
3    +I             // 実際の追加分
4    T     out      // ボックスパラメータ
5    NOP 0
```

加算が実行されます。RLO と BR ビットは変更されません。

C.3 パラメータ転送

ブロックのパラメータは、値として転送されます。ファンクションブロックでは、インスタンスデータブロックの実パラメータは、呼び出されたブロックの中で使用されます。ファンクションでは、現在値のコピーがローカルデータスタックに入れられます。ポインタはコピーされません。呼び出し前に、入力値はインスタンス DB または L スタックにコピーされます。呼び出し後に、出力値が変数にコピーされます。呼び出されたブロック内で操作対象となれるコピーは 1 つだけです。この操作に必要な STL 命令は、呼び出し側ブロック内にあるため、ユーザーから隠されていることに変わりはありません。

注記

メモリビット、入力、出力、またはペリフェラル I/O がファンクションの実アドレスとして使用されている場合は、他のアドレスとは異なる方法で扱われます。ここでは、L スタックを介さずに直接更新が実行されます。

例外:

対応する仮パラメータのデータタイプ BOOL の入力パラメータである場合、この現在のパラメータは L スタックを通して更新されます。

注意

呼び出されたブロックのプログラミング時には、出力として宣言されたパラメータも作成されていることを確認してください。作成されていない場合、出力の値は任意になります! ファンクションブロックでは、最後の呼び出しが指定したインスタンス DB からの値となり、ファンクションでは、L スタックに格納されている値になります。

以下の点を注意してください。

- 可能なら OUTPUT パラメータをすべて初期化します。
- Set および Reset の各命令を使用しないようにします。こうした命令は、RLO によって違ってきます。RLO の値が 0 の場合、任意の値が保たれます。
- ブロック内でジャンプする場合、出力パラメータが作成されたどの位置もスキップしないでください。BEC および MCR 命令の結果を忘れないでください。

索引

(

|

<

=

>

A

B

C

D

E

F

I

L

M

N

O

P

R

S

T

U

W

X

Z

あ

え

お

か

き

こ

さ

し

す

せ

そ

た

ち

て

と

は

ひ

ふ

へ

ほ

ま

め

ら

り

れ

ろ

わ