SIEMENS

SIMATIC

S7 S7-1200 プログラマブルコントローラ

システムマニュアル

製品概要	1
設置	2
PLC のコンセプト	3
デバイスコンフィグレーション	4
プログラミングのコンセプト	5
プログラミング手順	6
PROFINET	7
ポイントツーポイント(PtP) 通信	8
` '	9
通信	
通信 オンライン診断ツール	9

はじめに

法律上の注意

警告事項

本書には、ユーザーの安全性を確保し製品の損傷を防止するうえ守る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

回避しなければ、直接的な死または重傷に至る危険状態を示します。

小警告

回避しなければ、死または重傷に至るおそれのある危険な状況を示します。

⚠注意

回避しなければ、軽度または中度の人身傷害を引き起こすおそれのある危険な状況を示します(安全警告サイン付き)。

注意

回避しなければ、物的損傷を引き起こすおそれのある危険な状況を示します(安全警告サインなし)。

通知

回避しなければ、望ましくない結果や状態が生じ得る状況を示します(安全警告サインなし)。

複数の危険レベルに相当する場合は、通常、最も危険度の高い(番号の低い)事項が表示されることになっています。安全警告サイン付きの人身傷害に関する注意事項があれば、物的損傷に関する警告が付加されます。

有資格者

本書が対象とする製品 / システムは必ず有資格者が取り扱うものとし、各操作内容に関連するドキュメント、特に安全上の注意及び警告が遵守されなければなりません。有資格者とは、訓練内容及び経験に基づきながら当該製品 / システムの取り扱いに伴う危険性を認識し、発生し得る危害を事前に回避できる者をいいます。

シーメンス製品を正しくお使いいただくために

以下の事項に注意してください。

个警告

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。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

®マークのついた称号はすべて Siemens AG の商標です。本書に記載するその他の称号は商標であり、第三者が自己の目的において使用した場合、所有者の権利を侵害することになります。

免責事項

本書のハードウェアおよびソフトウェアに関する記述と、実際の製品内容との一致については検証済みです。 しかしなお、本書の記述が実際の製品内容と異なる可能性もあり、完全な一致が保証されているわけではありません。 記載内容については定期的に検証し、訂正が必要な場合は次の版て更新いたします。

はじめに

このマニュアルの目的

S7-1200 シリーズは、さまざまなオートメーションアプリケーションを制御できるプログラマブルロジックコントローラ(PLC)です。 コンパクトなデザインかつ低コストで、さまざまな命令セットを備えた **S7-1200** は、幅広い用途の制御に使用されます。 **S7-1200** モデルおよび Windows ベースのプログラミングツールには、オートメーションおけるさまざまな問題の解決に必要な柔軟性が備わっています。

このマニュアルには、S7-1200 PLC の設置およびプログラミングに関する情報が記載され、プログラマブルロジックコントローラに関する一般的な知識のあるエンジニア、プログラマ、設置工事担当者、電気技術者を対象として作成されています。

必要な基礎知識

このマニュアルを理解するには、オートメーションとプログラマブルロジックコントローラに関する一般的な知識が必要になります。

このマニュアルの適用範囲

このマニュアルはSTEP 7 Basic V10.5 およびS7-1200 製品群に適用されます。 このマニュアルで説明されているS7-1200 製品ファミリの完全なリストについては、技術仕様(ページ 391)を参照してください。

認定、CE ラベル、C-Tick、その他の規格

詳細については、技術仕様 (ページ 391)を参照してください。

サービスとサポート

さまざまな文書のほかに、インターネット上の下記のサイトにも、技術上のさまざまな 情報が用意されています。

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

技術的なご質問、トレーニングに関するお問い合わせ、**S7** 製品のご注文については、シーメンス正規販売店または弊社各支店にご相談ください。 担当者は、専門的な技術訓練を受け、ご使用の個別のシーメンス製品についての知識だけではなく、担当先での運用、処理、業務についての詳細な知識を身に着けていますから、どのような問題に対しても、迅速に効果的な答えを用意することができます。

目次

	はじめん	z	3
1	製品概要	Ę	11
	1.1	S7-1200 PLCについて	11
	1.2	シグナルボード	15
	1.3	シグナルモジュール	15
	1.4	通信モジュール	16
	1.5 1.5.1 1.5.2	STEP 7 Basic 用途に合わせた異なる表示方法 ヘルプ	17
	1.6	HMIパネル	22
2	設置		25
	2.1 2.1.1 2.1.2 2.1.3 2.1.4 2.1.5	取り付けおよび取り外しの手順 CPUの取り付けと取り外し シグナルモジュールの取り付けと取り外し 通信モジュールの取り付けおよび取り外し シグナルボードの取り付けおよび取り外し S7-1200 端子台コネクタの取り付けと取り外し	30 32 34
	2.2	配線のガイドライン	38
3	PLC のコ	コンセプト	43
	3.1 3.1.1 3.1.2 3.1.3 3.1.4 3.1.5	ユーザープログラムの実行 CPUの動作モード イベント実行の優先度およびキュー CPUメモリ S7-1200 CPUのパスワード保護 パスワードを忘れてしまった場合の復元	46 51 60 67
	3.2	データストレージ、メモリ領域、アドレス指定	69
	3.3	データタイプ	76
	3.4 3.4.1 3.4.2	メモリカードを使用する	82
	3.4.3 3.4.4	トランスファーカード	83

4	デバイン	スコンフィグレーション	91
	4.1	CPUを挿入する	92
	4.2	未指定のCPU用構成を検出する	93
	4.3	CPU の動作を構成する	94
	4.4	構成にモジュールを追加する	95
	4.5	モジュールのパラメータを構成する	96
	4.6	ネットワーク接続を作成する	98
	4.7	IPアドレスをプロジェクトで設定する	99
5	プログラ	ラミングのコンセプト	103
	5.1	PLCシステム設計用ガイドライン	103
	5.2	プログラムを構造化する	105
	5.3 5.3.1 5.3.2 5.3.3 5.3.4	ブロックを使用してプログラムを構造化するオーガニゼーションブロック(OB)ファンクション(FC)ファンクションブロック(FB)	108 110 111
	5.4	データの一貫性を理解する	114
	5.5	プログラム言語の選択	115
	5.6	コピー保護	117
	5.7	作成したプログラムの要素をダウンロードする	118
	5.8	作成したプログラムの要素をアップロードする	118
	5.9	プログラムをデバッグして試験する	120
6	プログラ	ラミング手順	121
	6.1.1.2 6.1.2 6.1.3 6.1.3.1 6.1.3.2 6.1.3.3 6.1.3.4 6.1.4 6.1.5	HSCのコンフィグレーション 比較 算術演算命令	
	6.1.5.1	MOD命令	152

6.1.6	ムーブ	. 160
6.1.6.1	スワップ命令	. 164
6.1.7	変換	. 165
6.1.7.1	スケーリング命令と正規化命令	. 168
6.1.8	プログラムコントロール	. 170
6.1.9	論理演算	. 172
6.1.10	シフトおよび回転	. 177
6.2	拡張命令	179
6.2.1	拡張命令用の共通のエラーパラメータ	_
6.2.2	クロックおよびカレンダ命令	
6.2.3	文字列命令と文字命令	
6.2.3.1	文字列データの概要	
6.2.3.2	文字列変換命令	
6.2.3.3	文字列操作命令	
6.2.4	プログラム制御命令	
6.2.4.1	スキャンサイクルウォッチドッグリセット命令	
6.2.4.2	スキャンサイクル停止命令	
6.2.4.3	エラーを取得する命令	
6.2.5	通信命令	
6.2.5.1	イーサネット通信を開く	. 212
6.2.5.2	ポイントツーポイント命令	
6.2.6	割り込み命令	
6.2.6.1	アタッチおよびデタッチ命令	
6.2.6.2	時間遅延割り込み命令の開始およびキャンセル	. 237
6.2.6.3	アラーム割り込み命令の無効化と有効化	240
6.2.7	PIDコントロール	. 241
6.2.8	モーションコントロール命令	
6.2.9	パルス命令	
6.2.9.1	CTRL_PWM命令	
6.3	グローバルライブラリ命令	248
6.3.1	USS	
6.3.1.1	USSプロトコルの使用要件	
6.3.1.2	USS_DRV命令	
6.3.1.3	USS_PORT命令	
	USS_RPM命令	
6.3.1.5	USS_WPM命令	
6.3.1.6	USSステータスコード	
6.3.2	MODBUS	
6.3.2.1	MB_COMM_LOAD	
6.3.2.2	MB_MASTER	
	MB_SLAVE	
PRUFIN	ET	
7.1	プログラミング装置との通信	. 295

7

	7.1.1	ハードウェア通信接続の確立	
	7.1.2	デバイスのコンフィグレーション	
	7.1.3	インターネットプロトコル(IP)アドレスの割り付け	
	7.1.3.1	IPアドレスのプログラミング装置とネットワークデバイスへの割り付け	
	7.1.3.2	IPアドレスのオンラインでの割り付け	
	7.1.3.3	IPアドレスをプロジェクトで設定する	
	7.1.4	PROFINETネットワークのテスト	
	7.2	HMIからPLCへの通信	
	7.2.1	HMIとCPU間の論理ネットワーク接続のコンフィグレーション	312
	7.3	PLCからPLCへの通信	313
	7.3.1	2 つのCPU間の論理ネットワーク接続のコンフィグレーション	
	7.3.2	送信パラメータと受信パラメータのコンフィグレーション	
	7.3.2.1	TSEND_C命令の送信パラメータのコンフィグレーション	
	7.3.2.2	TRCV_C命令の受信パラメータのコンフィグレーション	322
	7.4	リファレンス情報	327
	7.4.1	CPU上のイーサネット(MAC)アドレスの検索	
	7.4.2	ネットワークタイムプロトコル同期化のコンフィグレーション	329
8	ポイン	トツーポイント(PtP)通信	331
	8.1	RS232 およびRS485 通信モジュールの使用	332
	8.2	通信ポートのコンフィグレーション	333
	8.3	フロー制御の管理	335
	8.4	送信パラメータと受信パラメータのコンフィグレーション	337
	8.5	ポイントツーポイント通信のプログラミング	344
	8.5.1	ポーリングアーキテクチャ	345
	8.6	ポイントツーポイント命令	346
	8.6.1	ポイントツーポイント命令の一般的なパラメータ	
	8.6.2	PORT_CFG命令	
	8.6.3	SEND_CFG命令	351
	8.6.4	RCV_CFG命令	
	8.6.5	SEND_PTP命令	
	8.6.6	RCV_PTP命令	
	8.6.7	RCV_RST命令	
	8.6.8	SGN_GET命令	
	8.6.9	SGN_SET命令	
0	8.7	エラー イン診断ツール	
9	タンファ 9.1	1 ン診断ソール	
	9.2	オンラインで CPU に接続	381

	9.3	IPアドレスと時刻の設定	382
	9.4	オンラインCPUのCPUオペレータパネル	383
	9.5	サイクルタイムとメモリ使用率のモニタ	383
	9.6	CPUでの診断イベントの表示	384
	9.7	ユーザープログラムのモニタテーブル	385
Α	技術仕	策	391
	A.1	一般技術仕様	391
	A.2 A.2.1 A.2.2 A.2.3	CPU CPU 1211C仕様 CPU 1212C仕様 CPU 1214C仕様	398 404
	A.3 A.3.1 A.3.2 A.3.3	デジタルシグナルモジュール(SM) SM 1221 デジタル入力仕様 SM 1222 デジタル出力仕様 SM 1223 デジタル入出力仕様	419 420
	A.4 A.4.1	アナログシグナルモジュール(SM)SM 1231、SM 1232、SM 1234 アナログ仕様	
	A.5 A.5.1 A.5.2	シグナルボード(SB) SB 1223 2×24V DC入力 / 2×24V DC出力の仕様 SB 1232 1 アナログ出力仕様	437
	A.6 A.6.1 A.6.2	通信モジュール(CM) CM 1241 RS485 仕様 CM 1241 RS232 仕様	441
	A.7	SIMATICメモリカード	443
	A.8	入力シミュレータ	443
	A.9	I/O拡張ケーブル	445
В	電源容	量の計算	447
	B.1	電力要件の計算例	449
	B.2	電力要件の計算練習	450
С	注文番	号	451
	表さ		155

製品概要

1.1 S7-1200 PLC について

S7-1200 プログラマブルロジックコントローラ(PLC)には、オートメーションをサポートするさまざまなデバイスの制御に必要な柔軟性と能力が備わっています。 コンパクトなデザイン、柔軟性、構成、幅広い命令セットの組み合わせを備えた S7-1200 は、さまざまなアプリケーションの制御に最適です。

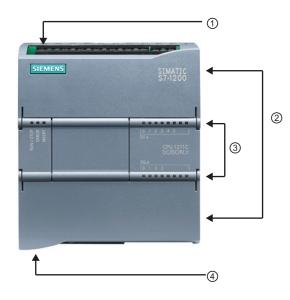
CPU モジュールには、マイクロプロセッサ、パワーサプライ、入力回路、出力回路が 組み込まれ、高性能で強力な PLC です。 プログラムをダウンロードすると、アプリケーションで使用されているデバイスの監視および制御に必要なロジックが CPU に保存されます。 CPU によって入力の監視が行われ、ユーザープログラムに基づいて出力されます。 ビット演算、カウント、タイミング、複雑な数学演算、インテリジェント機能を備えた機器との通信を行うことができます。

CPU および制御プログラムへのアクセスを保護する複数のセキュリティ機能が備わっています。

- どの CPU もパスワード保護機能を備え、CPU の各ファンクションにアクセスする ためのアクセス権限を設定することができます。
- 「ノウハウプロテクション」機能を使用して、特定のブロック内のプログラムを非表示にすることができます。 詳細については、「プログラミングコンセプト」 (ページ 117)を参照してください。

1.1 S7-1200 PLC について

CPU には PROFINET ポートが用意されていて、PROFINET 通信を行うことができます。 また、RS485 または RS232 ネットワークを通じて通信するための通信モジュールを使用することもできます。



- ① 供給電源端子
- ② 取り外し可能 I/O コネクタ(カバーの 裏側)
- ② メモリカードスロット(上部フロント カバーの裏)
- ③ オンボード I/O 状態表示 LED
- ④ PROFINET ポート(CPU 底面)

各CPUモデルには、アプリケーションに対する効果的なソリューションの作成に役立つ機能および能力が備えられています。 それぞれのCPUの詳細については、技術仕様 (ページ 391)を参照してください。

1.1 S7-1200 PLC について

特徴	CPU 1211C	CPU 1212C	CPU 1214C	
外形寸法(mm)	90 x 100 x 75		110 x 100 x 75	
ユーザーメモリ				
• ワークメモリ	• 25 KB		• 50 KB	
• ロードメモリ	• 1MB		• 2 MB	
• 保持メモリ	• 2 KB		• 2 KB	
オンボード I/O				
• デジタル	• 6入力/4出力	• 8入力/6出力	• 14 入力 / 10 出力	
• アナログ	2入力	2入力	2入力	
プロセスイメージサイズ	入力(I) 1024 バイト /	出力 (Q) 1024 バイト		
ビットメモリ(M)	4096 バイト		8192 バイト	
拡張モジュール数	なし	2	8	
シグナルボード	1			
通信モジュール	3 (左側拡張)			
高速カウンタ	3	4	6	
• 単相	• 100 kHz (3 チャ	,	• 100 kHz (3 チャンネ	
36-71	ンネル)	ネル) 30 kHz (1 チャンネ	ル) 30 kHz (3 チャンネル)	
• 差動	• 80 kHz (3 チャン	ル)	• 80 kHz (3 チャンネル)	
	ネル)	• 80 kHz (3 チャンネ	, ,	
	,	ル)		
		20 kHz (1 チャンネ		
		/レ)		
パルス出力	2			
メモリカード	SIMATIC メモリカード(オプション)			
リアルタイムクロック保持時間	標準 10 日 / 最小 6 日(40℃)			
PROFINET	イーサネット通信ポート(1 ポート)			
実数演算実行速度	18 μs / 命令			
ビット演算実行速度 0.1 μs / 命令				

1.1 S7-1200 PLC について

S7-1200 には、機能を拡張するためのさまざまなシグナルモジュールおよびシグナルボードが用意されています。 さらに、通信モジュールを追加することによって、他の通信プロトコルをサポートすることもできます。 それぞれのモジュールの詳細については、技術仕様 (ページ 391)を参照してください。

モジュール		入力	出力	入出力
シグナルモジュール(SM)	デジタル	DC 入力 8 点	DC 出力 8 点 リレー出力 8 点	DC 入力 8 点 / DC 出力 8 点 DC 入力 8 点 / リレー出力 8 点
		DC 入力 16 点	DC 出力 16 点 リレー出力 16 点	DC 入力 16 点 / DC 出力 16 点 DC 入力 16 点 / リレー出 力 16 点
	アナログ	アナログ入力 4点 アナログ入力 8点	アナログ出力 2 点 アナログ出力 4 点	アナログ入力 4 点 / アナロ グ出力 2 点
シグナルボー ド(SB)	デジタル	-	-	DC 入力 2 点 / DC 出力 2 点
	アナログ	-	アナログ出力 1 点	-

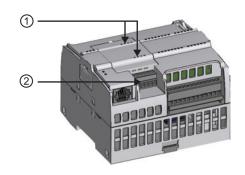
通信モジュール(CM)

- RS485
- RS232

1.2 シグナルボード

シグナルボード(SB)を使用して、CPU に I/O を追加することができます。 デジタル I/O またはアナログ I/O のどちらかの SB を 1 つ追加することができます。 SB は CPU の前面に接続します。

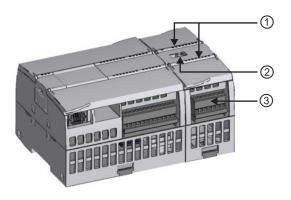
- デジタル I/O 4 点 SB (DC 入力 2 点 / DC 出力 2 点)
- アナログ出力 1 点 SB



- ① SB 上の I/O ステータス LED
- ② 取り外し可能 I/O コネクタ

1.3 シグナルモジュール

シグナルモジュールを使用して、CPU に機能を追加することができます。 シグナルモジュールは CPU の右側に接続します。



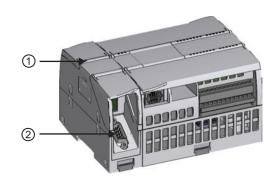
- ① シグナルモジュールの I/O ステータス LED
- ② バスコネクタ
- ③ 取り外し可能 I/O コネクタ

1.4 通信モジュール

1.4 通信モジュール

S7-1200 製品には、通信機能を追加するための通信モジュール(CM)が用意されています。 通信モジュールには、 RS232 および RS485 があります。

- CPU には最大3つの通信モジュールを連結することができます。
- 各 CM は、CPU (または他の CM)の左側に接続します。



- ① 通信モジュールステータス LED
- ② 通信コネクタ

1.5 STEP 7 Basic

STEP 7 Basic は、アプリケーションの制御に必要な論理回路の開発、編集、監視を行うためのユーザーフレンドリなソフトウェアです。PLC や HMI デバイスなど、プロジェクトに含まれるすべてのデバイスの管理および構成に使用するツールを提供します。STEP 7 Basic は、アプリケーションの制御プログラムを効率よく開発できるように、2種類のプログラミング言語(LAD および FBD)を提供します。また、プロジェクト内のHMI デバイスの作成および構成に使用するツールも提供します。

必要な情報が簡単に調べられるように、STEP 7 Basic には詳細なオンラインヘルプシステムが備わっていて、必要な情報を簡単に調べることができます。

STEP 7 Basic をインストールするには、コンピュータの DVD ドライブに DVD を挿入します。 インストールウィザードが自動的に開始され、画面の案内に従ってインストール作業を進めます。 詳細については、Readme ファイルを参照してください。

注記

Windows XP、または Windows Vista オペレーティングシステムを実行している PC 上に STEP 7 Basic ソフトウェアをインストールするには、管理者権限でログインする必要があります。

1.5.1 用途に合わせた異なる表示方法

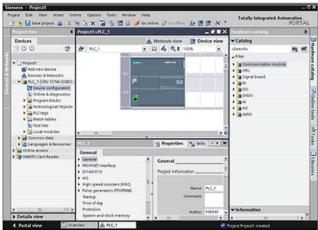
生産性を向上させるために、Totally Integrated Automation (TIA)ポータルツールセットには、2種類の表示方法が提供されています。 各ツールがワークフローに基づいて分類されているポータルビュー、またはプロジェクト内の要素で構成されているプロジェクトビューを使用することができます。 効率良く作業できる表示方式を選択することができます。 ポータルビューとプロジェクトビューをワンクリックで切り替えることができます。

ポータルビューでは、ハードウェアコンポーネントやネットワークの構成の作成など、実行するタスクに基づいてプロジェクトタスクの機能が表示され、ツールのファンクションがグループ化されています。

選択すべきタスクを簡単に決定 することができます。

プロジェクトビューでは、プロジェクト内のすべてのコンポーネントにアクセスすることができます。 すべてのコンポーネントが 1 つの画面に表示されるため、プロジェクトのどの要素にも簡単にアクセスすることができます。 表示されるプロジェクトには、作成または完了したすべての要素が含まれています。





1.5 STEP 7 Basic

1.5.2 ヘルプ

ヘルプにより簡単に調べられます

問題を短時間で解決するために、STEP 7 Basic には必要に応じたインテリジェント支援機能が備わっています。

- 入力フィールドには「ロールアウト」するヘルプが提供され、そのフィールドへの 正しい情報(有効な範囲やデータタイプ)の入力を支援します。 たとえば、無効な値 を入力しようとすると、メッセージテキストボックスがロールアウトして表示され、 有効な値の範囲が表示されます。
- インターフェースのツールチップには、詳細情報が続いて表示されるものがあります(命令の場合など)。 また、続いて表示されるツールチップには、オンライン情報システム(オンラインヘルプシステム)の特定のトピックにリンクしているものがあります。

さらに、STEP 7 Basic には、各 SIMATIC ツールの機能の説明を網羅した総合的な情報システムが備わっています。

ロールアウトヘルプおよび続いて表示されるツールチップ

さまざまなダイアログおよびタスクカードの入力フィールドでは、必要なデータ範囲やデータタイプをフィードバックするメッセージボックスが、ロールアウトして表示されます。

★Value outside the permissible range: [0..8191].

ソフトウェアインターフェースの各要素には、その要素の機能を説明するツールチップが備わっています。 [Open]アイコンや[Save]アイコンなど、詳細情報の不要な要素も存在します。 ただし、その要素についての詳細な説明を表示する機能を備えた要素も存在します。 この詳細情報は、ツールチップからのボックスに続いて表示されます (ツールチップの横に表示される黒色の三角形は、詳細情報が存在することを示しています)。

ソフトウェアインターフェースの項目の上 にカーソルを移動すると、その項目のツー ルチップが表示されます。 詳細情報を表示 するには、ツールチップの上にカーソルを 移動します。 続いて表示されたツールチッ プにも、関連トピックへのリンクが含まれ ているものがあります。 リンクをクリック すると、そのトピックが表示されます。 Network devices

Enables the
graphic
networking of
interfaces.

 Networking
devices in the
network view

情報システム

STEP 7 Basic には、オンラインの情報およびヘルプシステムが備わっていて、インストール済みの SIMATIC 製品のすべての情報を調べることができます。 この情報システムには、参照情報やさまざまな例も含まれています。 情報システムを表示するには、次のいずれかの手順を実行します。

- ポータルビューで[Start portal]を選択し、[Help]をクリックする。
- プロジェクトビューで、[Help]メニューの[Show help]を選択する。
- 表示されたツールチップのリンクをクリックし、そのトピックについての詳細情報 を表示する。

1.5 STEP 7 Basic

作業エリアの邪魔にならない場所に、情報システムのウィンドウが開きます。

情報システムの[Show/hide contents]ボタンをクリックすると、内容が表示され、ヘルプウィンドウが独立して表示されます。 ヘルプウィンドウのサイズを調節することができます。 [Contents]または[Index]タブを使用し、トピックまたはキーワードを指定して、情報システム内での検索を行うことができます。

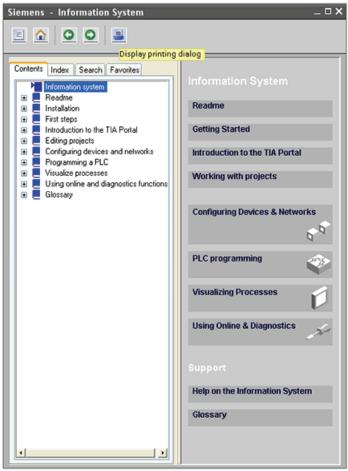


注記

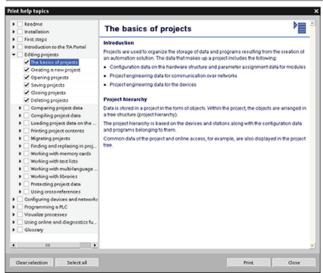
STEP 7 Basic の画面を最大化している場合は、[Show/hide contents]ボタンをクリックしても、ヘルプウィンドウが独立表示されません。 ヘルプウィンドウを独立して表示するには、[Restore down]ボタンをクリックします。 ヘルプウィンドウの移動およびサイズの調節を行うことができます。

情報システムからトピックを印刷する

情報システムで印刷を行うには、ヘルプウィンドウ上にある[Print]ボタンをクリックします。



情報システムで印刷を行うに は、ヘルプウィンドウ上にあ る[Print]ボタンをクリックし ます。



[Print]ダイアログを使用して、印刷するトピックを選択することができます。トピックが表示されていることを確認します。 任意のトピックを選択して印刷することができます。 [Print]ボタンをクリックすると、選択したトピックがプリンタに送信されます。

1.6 HMI パネル

1.6 HMI パネル

機械設計では視覚化コンポーネントが標準になっているため、SIMATIC HMI ベーシッ クパネルにはオペレータが基本的な制御および監視に使用するタッチスクリーンデバイ スが備わっています。 すべてのパネルは保護等級 IP65 に準拠し、CE、UL、cULus、 NEMA 4x の認定を取得しています。



KTP 400 Basic PN

- モノクロ(STN、グレースケール) 50 スクリーン
- 4インチタッチスクリーン(タッチ 200 アラーム キー4 点)
- 縦置きまたは横置き
- サイズ: 3.8"
- 解像度: 320×240

- 128 タグ

- **25** 種類のトレンド
- 32 KB のレシピメモリ
- 5種類のレシピ、20種 類のデータ記録、20種 類のエントリ



KTP 600 Basic PN

- カラー(TFT、256 色)または 50 スクリーン モノクロ(STN、グレースケ • 200 アラーム ール)
- 6インチタッチスクリーン (タッチキー6点)
- 縦置きまたは横置き
- サイズ: 5.7 インチ
- 解像度: 320×240

128 タグ

- 25 種類のトレンド
- 32 KB のレシピメモリ
- 5種類のレシピ、20種 類のデータ記録、20種 類のエントリ



KTP1000 Basic PN

- カラー(TFT、256 色)
- 10 インチタッチスクリー (タッチキー8点)
- サイズ: 10.4 インチ
- 解像度: 640×480

- 256 タグ
- 50 スクリーン
- 200 アラーム
 - **25** 種類のトレンド
 - 32 KB のレシピメモリ
 - 5種類のレシピ、20種 類のデータ記録、20種 類のエントリ

1.6 HMI パネル



TP1500 Basic PN

- カラー(TFT、256 色) 50 スクリーン
- 15 インチタッチスクリ 200 アラーム ーン
- サイズ: 15.1 インチ
- 解像度: 1024×768
- 256 タグ

 - 25 種類のトレンド
 - 32 KB のレシピメモリ (内蔵フラッシュ)
 - 5種類のレシピ、20種 類のデータ記録、20種 類のエントリ

1.6 HMI パネル

設置 2

S7-1200 機器は簡単に設置できるように設計されています。 **S7-1200** は、パネルまた は標準 **DIN** レールに取り付けることができます。 さらに **S7-1200** は、縦にも横にも取り付けることができます。 **S7-1200** は小型のため、スペースを効率良く使用することができます。

小警告

SIMATIC S7-1200 PLC はオープンタイプのコントローラです。 S7-1200 は、ハウジング、キャビネット、電気制御室内に格納する必要があります。 ハウジング、キャビネット、電気制御室内への格納は有資格者が行う必要があります。

下記の設置指針を遵守してください。遵守しないと、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

S7-1200 PLC の設置に際しては下記の要件を遵守してください。

S7-1200機器を高温、高電圧、電気的ノイズから避けて設置する

システムを構成する各装置を設置する際の一般的なルールとして、高電圧および大きな電気ノイズを発生する装置は、S7-1200 のような低電圧の論理装置から必ず離すようにします。

パネル内に S7-1200 を配置するとき、熱が発生するデバイスを考慮して、キャビネット内のなるべく低温の領域に、電子機器を配置するようにします。 高温の環境にさらさないようにすると、電子機器の寿命が延びます。

パネル内の配線の引き回しについても考慮します。 AC 電源配線やスイッチング周波数 の高い DC 配線と同じトレイに、低電圧信号の配線や通信ケーブルを収容することは避けるようにします。

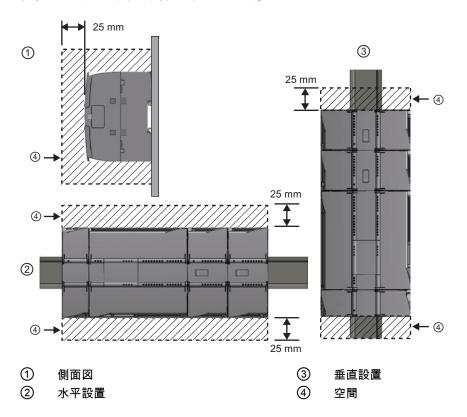
冷却および配線に必要な空間を確保する

S7-1200 の機器は自然空冷方式です。 適切な冷却が行われるように、この機器の上下 に 25 mm 以上の空間を確保する必要があります。 また、このモジュールの前面と筐体 の内側との空間を 25 mm 以上確保する必要があります。

/注意

縦向きに取り付ける場合は、最大許容周囲温度が 10℃低下します。縦向きにして取り付ける場合は、CPU がアセンブリの下側になるようにして S7-1200 システムを取り付けます。

S7-1200 システムのレイアウトを計画するとき、配線の引き回しおよび通信ケーブルの接続に必要な空間を十分に確保します。



電源容量

CPU には、CPU、シグナルモジュール、シグナルボード、通信モジュール、およびユーザー用 DC 24 V 電源が備えられています。

このCPUによって供給されるDC 5 V論理回路用電源、シグナルモジュール、シグナルボード、通信モジュールに必要なDC 5 V電源の要件については、技術仕様 (ページ 391)を参照してください。 所定の構成に関してCPUが供給する必要のある電源容量(または電流容量)を決定するには、「電源容量を計算する」 (ページ 447)を参照してください。

CPUには、DC 24 Vの入力点、シグナルモジュールのリレーコイル電源、他の要件用のDC 24 Vセンサ電源が備えられています。 DC 24V電源要件がセンサ電源の容量を超える場合は、DC 24V外部電源をシステムに追加する必要があります。 個別のS7-1200 CPUのD C 24 V センサ電源の容量については、技術仕様 (ページ 391)を参照してください。

DC 24 V 外部電源を追加する場合は、CPU のセンサ電源と並列に接続しないように注意してください。 電気ノイズからの保護を強化するために、それぞれの電源のコモン(M)を接続することを推奨します。

个警告

DC 24 V 外部電源を DC 24 V センサ電源と並列に接続すると、各電源がそれぞれの電位差を確立しようとして、2つの電源の競合が発生する可能性があります。

この競合によって両方の電源の寿命が短くなったり、すぐに故障する原因となり、その結果、PLCシステムの予測できない動作が発生することがあります。 予測がつかない動作によって、死傷事故などの重大な人的傷害や物的損害が発生するおそれがあります。

DC センサ電源と外部電源は、それぞれ異なる給電点への電源として使用する必要があります。

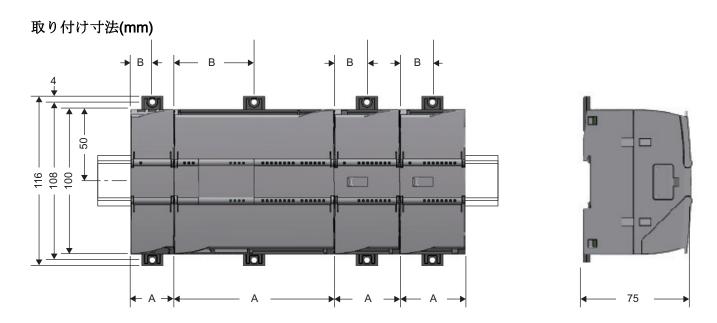
S7-1200 システムの DC 24 V 電源入力ポートには、共通の論理回路によって複数の M 端子が接続され、相互接続されているものがあります。 たとえば、データシートに「非絶縁」と記載されている場合に相互接続されている回路は次のとおりです。 CPU の DC 24 V 電源、SM のリレーコイル用電源入力、非絶縁アナログ入力用電源 すべての非絶縁 M 端子は、同じ外部基準電圧に接続する必要があります。

个警告

非絶縁 M 端子が複数の異なる基準電圧に接続されていると、予期しない電流が流れ、 PLC および接続されている装置の損傷や予測できない動作が発生する原因になること があります。

このガイドラインを遵守しなかった場合は、損傷や予測できない動作が発生する原因になり、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。 87-1200システムのすべての非絶縁 M 端子が、同一の電源の基準電圧に接続されていることを必ず確認してください。

2.1 取り付けおよび取り外しの手順



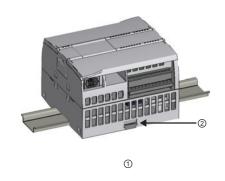
S7-1200 機器	幅A	幅 B	
CPU 1211C および CPU 1212C 9		90 mm	45 mm
	CPU 1214C	110 mm	55 mm
シグナルモジュール	8 点および 16 点 DC およびリレー入出力(8I、 16I、8Q、16Q、8I/8Q) アナログ入出力(4AI、8AI、4AI/4AQ、2AQ、 4AQ)	45 mm	22.5 mm
	16I/16Q リレー(16I/16Q)	70 mm	35 mm
通信モジュール:	CM 1241 RS232 および CM 1241 RS485	30 mm	15 mm

CPU、SM、CM を DIN レールまたはパネルに取り付けることができます。 レールに取り付ける場合は、DIN レールクリップでモジュールを固定します。 このクリップを引き出して突き出した位置にすると、ユニットをパネルに直付けするためのネジ取り付け位置になります。 デバイス上の DIN クリップの取り付け穴の内寸は 4.3 mm です。

自然空冷による冷却のために、ユニットの上下に **25 mm** の空間を確保する必要があります。

S7-1200 デバイスの取り付けと取り外し

CPU は標準 DIN レールまたはパネルに簡単に取り付けることができます。 DIN レールには デバイスを固定するための DIN レールクリップが備わっています。 また、このクリップを 引き出して突き出した位置にすると、ユニットをパネルに取り付けるためのネジ取り付け位置になります。





- ① DIN レールと取り付け
- ③ パネルへの取り付け 対置のクリ ④ パネル取り付け時の!
- ② DIN レール取り付け固定位置のクリップ
- ④ パネル取り付け時の突き出した位置のクリップ

電気装置の取り付けおよび取り外しを行う前に、その装置の電源がオフになっていることを必ず確認します。 また、関連機器の電源もオフになっていることを確認しておく必要があります。

<u>!</u>警告

S7-1200 および電源が供給されている関連機器の取り付けおよび取り外しによって、感電または予測できない装置の動作の原因になることがあります。

取り付けおよび取り外し作業中は、S7-1200 および関連機器のすべての電源をオフにしていないと、感電または予測できない装置の動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

安全上の注意事項を遵守し、S7-1200 CPU または関連機器の取り付けおよび取り外しを行う前に、S7-1200 の電源を必ずオフにしてください。

S7-1200 デバイスの交換または取り付けを行うときは、適正なモジュールまたは等価のデバイスを必ず使用してください。

个警告

S7-1200 モジュールを正しく取り付けないと、**S7-1200** のプログラムに予測できない動作が発生することがあります。

交換した S7-1200 機器のモデル、方向、順序が元のものと異なると、予測できない装置の動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。 S7-1200 機器を交換する場合は、元のモデルと同じモデルを使用し、正しい向きと正しい位置に取り付けてください。

2.1 取り付けおよび取り外しの手順

2.1.1 CPUの取り付けと取り外し

設置

CPU はパネルまたは DIN レールに取り付けることができます。

注記

必要な通信モジュールを CPU に連結してから、1 つのユニットとして取り付けます。 シグナルモジュールは、CPU を取り付けた後で別に取り付けます。

CPU をパネルに取り付けるには次の手順を実行します。

- 1. 取り付け寸法を参照して取り付け穴の位置を決め、穴を開けてねじを切ります(M4 または米国標準規格番号 8)。
- 2. モジュールから取り付けクリップを引き出します。 CPU の上面と底面の両方の DIN レールクリップが引き出された位置になっていることを確認します。
- 3. クリップの穴にねじを使用して、パネルにモジュールを固定します。

注記

システムを振動の多い環境に取り付ける場合または縦に取り付ける場合は、S7-1200 をパネルに取り付けると保護を強化することができます。

CPU を DIN レールに取り付けるには次の手順を実行します。





- 1. DIN レールを取り付けます。 パネルへのレールの取り付け間隔は 75 mm にします。
- 2. DIN レールの上側に CPU を引っ掛けます。
- 3. CPU がレールに密着するように、CPU 底部の DIN レールクリップを引き出します。
- 4. CPU を回転させるようにしてレールにはめ込みます。
- 5. クリップを押し込んで CPU をレールに固定します。

取り外し

CPU を取り外す前に CPU の電源をオフにし、I/O コネクタ、配線、ケーブルを CPU から切り離します。 CPU と連結されている通信モジュールを 1 つのユニットとして取り外します。 シグナルモジュールは取り付けたままにしておきます。







CPU にシグナルモジュールが接続されている場合は、バスコネクタを引っ込めます。

- 1. シグナルモジュール上部のタブにスクリュードライバを当てます。
- 2. ドライバを押し込んで、CPU からコネクタを切り離します。
- 3. タブは右端まで完全にスライドさせます。

CPU を取り外します。

- 1. DIN レールクリップを引き出して、レールと CPU の固定を解除します。
- 2. CPU を回転させるようにして持ち上げてレールから外し、システムから CPU を取り外します。

2.1 取り付けおよび取り外しの手順

2.1.2 シグナルモジュールの取り付けと取り外し

設置

SM の取り付けは CPU を取り付けた後で行います。





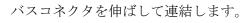
CPU の右側からコネクタ用のカバーを取り外します。

- カバーの上側にあるスロットにスクリュードライバを当てます。
- 慎重にカバーの上部をこじ開けて取り外します。 カバーは再使用に備えて保管しておきます。



CPU の隣に SM を並べます。

- 1. DIN レールの上側に SM を引っ掛けます。
- 2. SM をレールに密着させることができるように、 下側の DIN レールクリップを引き出します。
- 3. SM を回転させるようにして CPU の隣に正しく はめ込み、下側のクリップを押し込んで、SM を レールに固定します。



- 1. SM の上部のタブにスクリュードライバを当てます。
- 2. タブを左端の位置まで完全にスライドさせて、バスコネクタを CPU に押し込みます。

バスコネクタを押し込むことにより、SM の機械的 連結および電気的接続が行われます。





シグナルモジュールとシグナルモジュールの連結も同じ手順で行います。

取り外し

SM の取り外しは、CPU を取り外したりせず、他の SM もそのままにした状態で行うことができます。 CPU を取り外すには、CPU の電源をオフにし、I/O コネクタ、配線、ケーブルを CPU から切り離す必要があります。

バスコネクタを引っ込めます。

- 1. SM の上部のタブにスクリュー ドライバを当てます。
- 2. ドライバを押し込んで、CPU からコネクタを切り離します。
- 3. タブは右端まで完全にスライド させます。





さらに右側に別のSMが存在する場合は、そのSMでもこの手順を繰り返します。

SM を取り外します。

- 2. SM を回転させるようにして持ち上げ、レールから外します。 システムから SM を取り外します。
- 3. 必要に応じて、CPU のバスコネクタが汚れないようにカバーしておきます。



シグナルモジュールとシグナルモジュールの取り外しも同じ手順で行います。

2.1 取り付けおよび取り外しの手順

2.1.3 通信モジュールの取り付けおよび取り外し

設置

CM を CPU に連結し、1 つのユニットとして DIN レールまたはパネルに取り付けます。

CPU の左側からバスカバーを取り外します。

- バスカバーの上側にある スロットにスクリュード ライバを挿入します。
- 2. 慎重にカバーの上部をこ じ開けて外します。





バスカバーを取り外します。カバーは再使用に備えて保管しておきます。

ユニット同士を連結します。

- バスコネクタの位置および CM の凸部と CPU の 凹部を合わせます。
- 2. ユニット同士を押し付け あって、凸部を凹部には め込みます。





ユニットを DIN レールまたはパネルに取り付ける

- 1. DIN レールに取り付ける場合は、CPU と CM の両方の上側のレールクリップがラッチ(押し込まれた)位置になっていて、下側のレールクリップが引き出された位置になっていることを確認します。
- 2. CPUとCMのユニットを CPUの取り付けおよび取り外レ (ページ 30)と同様の手順で取り付けます。
- 3. DIN レールにデバイスを取り付けたら、下側のレールクリップをラッチ位置に押し込んで、デバイスを DIN レールに固定します。

パネルに取り付ける場合は、上下の DIN レールクリップが引き出された位置になっていることを確認します。

取り外し

CPUとCMを1つのユニットとして、DINレールまたはパネルから取り外します。



CM を取り外す準備をします。 1. CPU の電源を切ります。

- 2. CPUとCMから、I/Oコネクタおよびすべての配 線ケーブルを切り離します。
- 3. DIN レールに取り付けられている場合は、CPU と CM の下側の DIN レールクリップを引き出した 位置にします。
- 4. DIN レールまたはパネルから CPU と CM を一体 で取り外します。



CM を取り外します。

- 1. CPUとCMをしつかりと掴みます。
- 2. 両者を引き離します。

モジュールを分離する際、道具を使用するとユニッ トが損傷するため、道具は使用しないでください。

2.1 取り付けおよび取り外しの手順

2.1.4 シグナルボードの取り付けおよび取り外し

設置

CPU の電源を切断し、CPU から上面と底面の端子台カバーを取り外して、SB を取り付ける準備をします。

SB を取り付けるには次の手順を実行します。

- CPU の上面のカバーの後ろ側に あるスロットにスクリュードラ イバを挿入します。
- 2. 慎重にカバーをこじ開けて、 CPU から取り外します。
- 3. CPU の上部の取り付け位置に SB をまっすぐに挿入します。
- 4. スナップがはまる位置まで SB を押し込みます。
- 5. 端子台カバーを元どおりに取り 付けます。







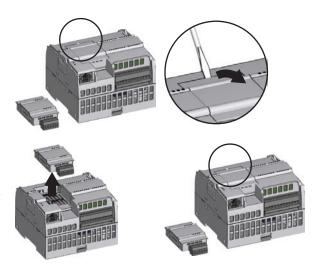


取り外し

CPU の電源を切り、CPU から上面と底面の端子台カバーを取り外して、SB を取り外 す準備をします。

SB を取り外すには次の手順を実行します。

- SBの上部のスロットにスクリュードライバを挿入します。
- 2. 慎重に SB をこじ開けて、CPU から取り外します。
- CPU の上部の取り付け位置から 真上に SB を引き抜きます。
- 4. SB カバーを元どおりに取り付けます。
- 5. 端子台カバーを元どおりに取り 付けます。



2.1.5 S7-1200 端子台コネクタの取り付けと取り外し

CPU、SB、SM モジュールには、配線が簡単に接続できるように、取り外し可能 I/O コネクタが備えられています。 端子台コネクタを取り外す準備

- CPU の電源を切ります。
- コネクタの上側のカバーを開きます。

コネクタを取り外すには次の手順を実行します。

- 1. コネクタの上部にあるスクリュードライバの先を挿入するスロットを探します。
- そのスロットにスクリュードライバを挿入します。
- 3. 慎重にコネクタの上部をこじ開けて、CPU から外します。 コネクタのスナップが外れます。
- 4. コネクタを掴んで CPU から取り 外します。





コネクタを取り付けるには次の手順を実行します。

- 1. CPU から電源を切り離し、端子台のカバーを開いて、端子台を取り付ける準備をします。
- 2. コネクタとユニット上のピンの位置を合わせます。
- 3. コネクタの配線側の端とコネクタベースのリムの内側の位置を合わせます。
- 4. 強く押し込みながらコネクタを回転させるようにして、スナップをはめ込みます。

コネクタが正しい位置に完全にはめ込まれていることを確認 します。





2.2 配線のガイドライン

2.2 配線のガイドライン

すべての電気装置の適切な接地および配線は、システムの最適運用を可能にし、アプリケーションおよびS7-1200の電気ノイズに対する保護を強化するのに役立ちます。S7-1200の配線図については、技術仕様 (ページ 391)を参照してください。

前提条件

電気装置の接地および配線を行う前に、その装置の電源がオフになっていることを必ず 確認します。 また、関連機器の電源もオフになっていることを確認しておく必要があ ります。

S7-1200 および関連機器の配線に際しては、適用される電気関連法規のすべてを遵守する必要があります。 適用される法令および条例のすべてを遵守して、設置および運用するようにします。 適用される法令および基準については、監督官庁にお問い合わせください。

小警告

S7-1200 および電源が供給先の関連機器の取り付けおよび配線が、感電や予測できない装置の動作の原因になることがあります。 取り付けおよび取り外し作業中は、S7-1200 および関連機器のすべての電源を切断していないと、感電または予測できない装置の動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

安全上の注意事項を遵守し、S7-1200 または関連機器の取り付けおよび取り外しを行う前に、S7-1200 の電源が切断されていることを必ず確認してください。

S7-1200 システムの接地および配線の設計に際しては、常に安全を考慮する必要があります。 S7-1200 などの電子制御装置は、制御および監視を行っている設備の誤操作や予測できない動作の原因になることがあります。 そのため、潜在的な人的傷害や物的損害の発生を防止するために、S7-1200 から独立した安全装置を実装する必要があります。

个警告

制御装置が安全でない状態に陥り、被制御機器に予測できない動作が発生することがあります。 このような予測できない動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

S7-1200 から独立した、緊急停止装置、電子機械的補助装置、冗長安全装置などを使用する必要があります。

絶縁のためのガイドライン

S7-1200 AC 電源境界と AC 回路に対する I/O の境界は、AC 電源電圧と低電圧回路が安全に分離されているように設計および認定されています。 これらの境界の絶縁には、さまざまな規格に基づいて、二重または強化、つまり、基本と補完の絶縁が含まれています。 オプティカルカプラ、コンデンサ、トランス、リレーなど、これらの境界に渡って使用されるコンポーネントは、安全な絶縁が得られるものとして認定を取得しています。 これらの要件を満たす必要がある分離境界は、S7-1200 製品データシートに、1500 VAC 以上の絶縁特性があるものとして表示されています。 これは、承認されている方法に基づいて実施された(2Ue + 1000 VAC)または等価の工場試験の結果に基づくものです。 S7-1200 安全分離境界は 4242 VDC の型式試験を受けています。

電源を内蔵している S7-1200 のセンサ電源出力、通信回路、内部論理回路への電源は、EN 61131-2 に基づく SELV (安全な超低電圧)として供給されます。

S7-1200 の低電圧回路の安全特性を保持するために、通信ポート、アナログ回路、すべての公称 24 V 電源、I/O 回路への外部接続は、さまざまな規格に基づいて、SELV、PELV、Class 2、Limited Voltage (電圧制限)、または Limited Power (電力制限)の要件に適合する承認済み電源から電源を供給する必要があります。

/ 警告

AC 電源から低電圧を供給するために非絶縁または単絶縁電源を使用すると、通信回路や低電圧センサ配線など、触れても安全であると考えられている回路に危険な電圧が発生する可能性があります。

このような予測できない高電圧による感電によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

触れても安全な電圧制限回路を備えた電源として承認されている降圧コンバータ以外 は使用しないようにしてください。

S7-1200 の接地用ガイドライン

アプリケーションの接地を行うための最善の方法は、**S7-1200** とその関連機器のすべてのコモン接続および接地接続を単一点に接地する方法です。 この単一点をシステムの接地グラウンドに直接接続します。

接地配線はできるだけ短くし、2 mm² (14 AWG)などの太い線を使用します。

接地点を決定するときは、安全な接地のための要件と保護遮断器の適切な動作を考慮します。

2.2 配線のガイドライン

S7-1200 の配線用ガイドライン

S7-1200 の配線を設計する際は、単一の切断スイッチで、S7-1200 CPU の電源、すべての入力回路、およびすべての出力回路の電源を同時に切断できるようにします。 電源配線の異常電流を遮断するために、ヒューズやサーキットブレーカなどの過電流保護装置を設けます。 それぞれの出力回路に、ヒューズや電流制限器を配置して、保護機能を高めることを考えます。

雷によるサージ電圧にさらされる可能性のあるすべての配線にサージ保護器を組み込みます。

AC 電源配線やスイッチング周波数の高い DC 配線と同じトレイに、低電圧信号を伝送する配線や通信ケーブルを収容することは避けるようにします。 信号配線をニュートラル線またはコモン線と対にするなど、常に線を対にして配線します。

配線の長さはできるだけ短くし、必要な電流容量を満たす太さのものを使用します。 コネクタには、2 mm²~0.3 mm² (14 AWG~22 AWG)の太さの電線を接続することができます。 電気ノイズに対する保護が必要な部分には、シールド付きケーブルを使用します。 通常は、S7-1200 でシールドを接地することで最良の結果が得られます。

外部電源によって電源を供給している入力回路の配線の場合は、その回路に過電流保護器を挿入します。 S7-1200 の DC 24 V センサ電源から電源を供給している回路の場合、センサ電源に電流制限機能が備わっているため、過電流保護器は不要です。

すべての S7-1200 モジュールには、取り外し可能 I/O コネクタが備わっています。 コネクタ部での接触不良を防ぐために、コネクタを確実に固定し、配線をコネクタに確実に接続します。 コネクタの損傷を防ぐために、ネジを締め過ぎないように注意する必要があります。 コネクタのネジの最大締め付けトルクは 0.56 N-m (5 インチポンド)です。

望ましくない電流が流れ込むのを防ぐために、S7-1200 の所定のポイントに絶縁境界が設けられています。 システムの配線を計画する際に、これらの絶縁境界を考慮する必要があります。 絶縁境界の絶縁定格と分離ポイントについては技術仕様を参照してください。 安全な境界としては、1500 VAC 以上の定格の絶縁境界を確立する必要があります。

誘導負荷についてのガイドライン

誘導負荷の場合、制御出力をオフにしたときの電圧の上昇を抑制するために、抑制回路を備えている必要があります。 抑制回路を使用することによって、誘導負荷をオフにしたときの高電圧による早期故障から出力を保護します。 また、抑制回路によって、誘導負荷のスイッチングによって発生する電気ノイズも低減します。 物理的にできるだけ負荷の近くに負荷と電気的に並列に外部抑制回路を接続すると、最も効果的に電気ノイズを減らすことができます。

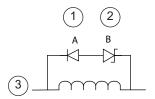
注記

抑制回路の効果はアプリケーションによって異なりますから、アプリケーションごとに 検証する必要があります。 抑制回路に使用するすべてのコンポーネントが、アプリケーションに必要な定格を満たしていることを確認します。

DC 誘導負荷の制御

S7-1200 DC 出力には、通常のアプリケーションにおける誘導負荷に適した抑制回路が使用されています。 DC または AC 負荷にはリレーが使用できるようになっているため、内部保護装置は備わっていません。 DC 負荷に対する抑制回路の例を図に示します。

通常のアプリケーションでは、誘導負荷と並列にダイオード(A)を追加するだけで十分ですが、ターンオフ時間を短くする必要がある場合は、ツェナダイオード(B)を追加することをお勧めします。

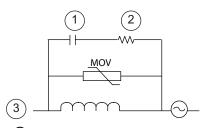


- ① 1N4001 ダイオードまたは同等品
- ② 8.2 V ツェナダイオード(DC 出力)、36 V ツェナダイオード(リレー出力)
- ③ 出力点

出力電流の大きさに適合しているツェナダイオードを選択する必要があります。

AC 負荷を制御するリレー出力

AC 115 V / 230 V 負荷の制御にリレーを使用する場合は、図に示すように、AC 負荷と並列に抵抗とコンデンサで構成したネットワークを接続します。 ピーク電圧を抑制するために酸化金属バリスタ(MOV)を使用することもできます。 定格電圧が公称ライン電圧より 20 %以上高い MOV を選択してください。



- ① 0.1 µF
- ② $100 \sim 120 \Omega$
- ③ 出力点

2.2 配線のガイドライン

ランプ負荷についてのガイドライン

ランプ負荷は突入電流が大きく、リレーの接点が損傷します。 この突入電流は、タングステンランプの場合で、定常電流の 10~15 倍にも達します。 アプリケーションのライフタイムにおいて頻繁にオンオフするランプ負荷には、交換可能な中間リレーまたはサージリミッタを使用することをお勧めします。

PLC のコンセプト

3.1 ユーザープログラムの実行

プログラムを効率良く構造化することのできる、次のタイプのプログラムブロックが CPU によってサポートされています。

- オーガニゼーションブロック(OB)はプログラムの構造を定義します。 動作および開始イベントが定義済みの OB も存在しますが、独自の開始イベントを定義した OB を作成することもできます。
- ファンクション(FC)およびファンクションブロック(FB)には、特定のタスクまたはパラメータの組み合わせに対応したプログラムコードが含まれています。 それぞれの FC または FB には、呼び出し元のブロックとデータを共有するための入力と出力のパラメータのセットが提供されています。 また、FB は、関連のデータブロック(インスタンス DB)を使用して実行間の値の状態を保持し、それをプログラム内の他のブロックが使用することができます。
- データブロック(DB)には、プログラムブロックが使用することのできるデータが格納されます。

ユーザープログラムの実行は、RUN モードに入ったときに 1 回実行されるオプションのスタートアップオーガニゼーションブロックによって始まり、引き続き 1 つ以上のプログラムサイクル OB が周期的に実行されます。 OB は割り込みイベントに関連付けることもできます。割り込みイベントは、標準イベントまたはエラーイベントのどちらでも可能で、関連付けられている標準イベントまたはエラーイベントが発生すると実行されます。

ファンクション(FC)またはファンクションブロック(FB)は、次のレベルまで、OB または他の FC または FB から呼び出すことのできるプログラムコードのブロックです。

- プログラムサイクルまたはスタートアップ OB からは 16 レベル
- 遅延割り込み、周期割り込み、タイムエラー割り込み、または診断エラー割り込みの OB からは 4 レベル

FB はデータブロック(DB)に直接関連付けられていますが、FC は特定のどの DB にも関連付けられていません。FB は、パラメータの受け渡しおよび中間値と結果の保存に DB を使用します。

ユーザープログラム、データ、コンフィグレーションのサイズは、CPU 内のロードメモリおよびワークメモリの使用可能な容量によって制限されます。 使用可能なワークメモリ内でサポートできるブロックの数は無制限です。

各サイクルには、出力の書き込み、入力の読み取り、ユーザープログラム命令の実行、およびシステムメンテナンスまたはバックグラウンド処理が含まれます。 このサイクルは、スキャンサイクルまたは単にスキャンと呼ばれます。

シグナルボード、シグナルモジュール、および通信モジュールの検出は電源投入時にのみ行われ、これらへのログインが行われます。

注記

電源をオンにしたままでのシグナルボード、シグナルモジュール、および通信モジュールの挿入(ホットスワップ:活線挿抜)はサポートされていません。 ただし、SIMATIC メモリカードは例外で、CPU の電源がオンの状態で脱着することができます。

デフォルトの構成では、すべてのデジタルおよびアナログの I/O ポイントは、プロセスイメージと呼ばれるメモリ領域を使用して、スキャンサイクルに同期して更新されます。 プロセスイメージには、物理的な入出力(CPU 上の物理的な I/O ポイント、シグナルボード、およびシグナルモジュール)の情報が格納されています。

CPU は次のタスクを実行します。

- CPUは、プロセスイメージ出力領域から物理的な出力に出力を書き込みます。
- CPU は、ユーザープログラムを実行する直前に物理的な入力から入力を読み取り、その値をプロセスイメージの入力領域に格納します。 これによって、ユーザー命令実行中の値の一貫性が保証されます。
- CPU は、ユーザー命令のロジックを実行し、実際の物理的な出力に書き込むのではなく、 プロセスイメージの出力領域の出力値を更新します。

この処理によって、所定のサイクルのユーザー命令の実行中のロジックの一貫性を保持する とともに、物理的な出力ポイントの変動を防止し、プロセスイメージ出力領域の状態が頻繁 に変化するのを防いでいます。

プロセスイメージに格納するデジタルおよびアナログの I/O ポイントを指定することができます。 デバイスビューにモジュールを挿入した場合、そのデータは S7-1200 CPU のプロセスイメージに配置されます(デフォルト)。 モジュールとプロセスイメージ領域間のデータ交換は、プロセスイメージの更新時に、CPU によって自動的に処理されます。 プロセスイメージの自動更新からデジタルまたはアナログの I/O ポイントを削除するには、削除するデバイスを[Device configuration]で選択して[Properties]タブを表示し、必要に応じて、削除する I/O ポイントを特定するために展開し、[IO addresses/HW identifier]を選択します。

[Process image]のエントリを [Cyclic PI]から[---]に変更します。 プロセスイメージの自動更 新から削除したポイントを元に戻すには、エントリの選択を[Cyclic PI]に戻します。

命令の実行時に直ちに物理的な入力値の読み取りおよび物理的な出力値の書き込みを行うことができます。 即時読み取りでは、物理的な入力の現在の状態へのアクセスが行われ、そのポイントがプロセスイメージに格納するように構成されていても、プロセスイメージ入力領域の更新は行われません。 物理的な出力への即時書き込みでは、プロセスイメージ出力領域と(そのポイントがプロセスイメージに格納するように設定されている場合)、物理的な入力の両方の更新が行われます。 プログラムで、プロセスイメージを使用せずに物理的なポイントに対して直接 I/O データの即時アクセスを行う場合は、I/O アドレスの後に「:P」を付加します。

スタートアップパラメータを構成する

電源投入サイクルの後の CPU のスタートアップを構成するには CPU のプロパティを使用します。



STOP モード、RUN モード、STARTUP モ ード(電源投入サイクル の前)のどのモードで CPU をスタートさせ るかを選択します。

CPU は RUN モードに入る前にウォームリスタートを実行します。 ウォームリスタートでは、すべての非保持メモリがデフォルト値にリセットされますが、保持メモリに格納されている現在の値は保持されます。

注記

CPU は、ダウンロード後には常にリスタートを実行します。

プロジェクトの要素(プログラムブロック、データブロック、ハードウェア構成など)を1つでもダウンロードすると、CPU は次回の RUN モードへの移行時に必ずリスタートを実行します。 リスタートでは、入力のクリア、出力の初期化、非保持メモリの初期化に加え、保持メモリ領域の初期化も行われます。

ダウンロードに続けて実行されるリスタートの後は、以降のすべての STOP から RUN への移行時にウォームリスタートが実行されます(保持メモリの初期化は行われません)。

3.1.1 **CPU**の動作モード

CPU には、3 つのオペレーティングモードがあります。 「STOP モード」、「STARTUP モード」、「RUN モード」の 3 種類の動作モードがあります。 CPU の前面のステータス LED によって、現在の動作モードが表示されます。

- STOP モードでは、CPU はプログラムを実行していないため、プロジェクトをダウンロードすることができます。
- STARTUP モードでは、スタートアップ OB (存在する場合)が 1 回だけ実行されます。 RUN モードのスタートアップ段階では、割り込みイベントの処理は行われません。
- RUN モードでは、スキャンサイクルが反復して実行されます。 プログラムサイク ルのどの時点でも割り込みイベントの発生および処理が可能です。

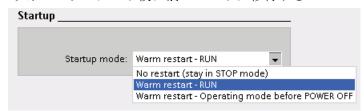
RUN モードの間はプロジェクトのダウンロードを実行することはできません。

CPU はウォームリスタートによる RUN モードへの移行をサポートしています。 ウォームリスタートにはメモリのリセットは含まれません。 保持システムでないすべてのシステムおよびユーザーデータはウォームリスタートによって初期化されます。 保持メモリのユーザーデータは保持されます。

メモリリセットは、すべてのワークメモリをクリアし、保持および保持でないメモリ領域をクリアして、ロードメモリをワークメモリにコピーします。 メモリリセットでは、診断バッファおよび IP アドレスの値はクリアされません。

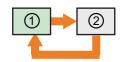
プログラミングソフトウェアを使用して、リスタート時に CPU の電源投入モードを指定することができます。 この構成項目は、この CPU の[Device Configuration]の下の [Startup]の下に表示されます。 電源を投入すると、CPU は電源投入診断とシステムの 初期化を実行します。 次に、CPU は所定の電源投入モードに移行します。 検出された エラーによって CPU が RUN モードに移行できないことがあります。 CPU は次の電源投入モードをサポートしています。

- STOP モード
- ウォームリスタート後に RUN モードに移行する
- ウォームリスタート後に前のモードに移行する



プログラミングソフトウェアのオンラインツールで STOP または RUN コマンドを使用することによって、現在の動作モードを変更することができます。 プログラム内に STP 命令を記述して、CPU の動作モードを STOP モードに変更することもできます。 このようにすれば、プログラムロジックに基づいて、プログラムの実行を停止することができます。

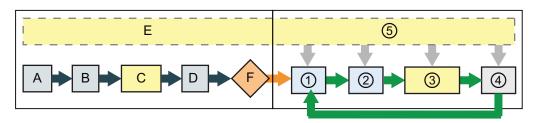
STOP モードでは、**CPU** が通信要求を処理し(必要に応じて) ①、自己診断を実行します ②。



STOP モードでは、CPU がユーザープログラムの実行を停止するため、プロセスイメージの自動更新は発生しません。

プロジェクトのダウンロードができるのは、CPU が STOP モードのときだけです。

RUN モードでは、CPU は次の図に示すタスクを実行します。



STARTUP

- A Iメモリ領域をクリアする
- B 直前の値または代替値のどちらかを使 用して出力を初期化する
- C スタートアップ OB を実行する
- D 物理的な入力の状態をIメモリにコピー する
- E 割り込みイベントが発生した場合は、 RUN モードで処理するために、キュー に格納する
- F 物理的な出力への Q メモリの書き込み を可能にする

RUN

- ① Qメモリを物理的な出力に書き込む
- ② 物理的な入力の状態を I メモリにコピーする
- ③ プログラムサイクル OB を実行する
- ④ 自己診断を実行する
- ⑤ スキャンサイクルの各部で割り込みおよび通信を処理する

STARTUP 処理

動作モードが STOP モードから RUN モードに変化すると、CPU はプロセスイメージ 入力をクリアし、プロセスイメージ出力を初期化して、スタートアップ OB を処理します。 スタートアップ OB 内の命令が使用するプロセスイメージ入力の値は、現在の物理的な入力値ではなく 0 になります。 したがって、STARTUP モードで物理的な入力の現在の状態を読み取るには、即時読み取りを実行する必要があります。 スタートアップ OB および関連の FC および FB が次に実行されます。 複数のスタートアップ OB が存在する場合は、OB 番号の最も小さいものから、OB 番号の順序で、それぞれが実行されます。

スタートアップ OB のそれぞれに、保持データおよび時計の有効性を調べるのに役立つ情報が含まれています。 これらのスタートアップ値を調べて、対応する操作を実行する命令をスタートアップ OB 内にプログラムすることができます。 スタートアップ OB では、次のスタートアップロケーションがサポートされています。

入力	データタ イプ	説明
LostRetentive	BOOL	保持データ保存領域が失われた場合にこのビットが 1 になります
LostRTC	BOOL	時計(リアルタイムクロック)が失われた場合にこのビット が 1 になります

また、スタートアップ処理では、CPU は次のタスクも実行します。

- スタートアップ中は、割り込みはキューに格納され、処理されません。
- スタートアップ中は、サイクルタイムの監視は行われません。
- スタートアップ時に、HSC (高速カウンタ)、PWM (パルス幅変調)、PtP (ポイント ツーポイント通信)モジュールへの構成変更を行うことができます。
- 実際の HSC、PWM、PtP 通信モジュールの操作は、実行時にのみ発生します。

CPU は、スタートアップ OB の実行が終了すると RUN モードに移行し、連続スキャンサイクル内の制御タスクを処理します。

RUN モードでスキャンサイクルを処理する

各スキャンサイクルでは、CPU は出力の書き込み、入力の読み取り、ユーザープログラムの実行、通信モジュールの更新、内部準備作業の実行、ユーザー割り込みイベントおよび通信要求への応答を実行します。 通信要求は、スキャン中を通して定期的に処理されます。

これらの操作(ユーザー割り込みイベントを除く)は、定期的に順次に行われます。 有効なユーザー割り込みイベントのサービスは、発生した順序で優先度に基づいて行われます。

システムによって、最大サイクルタイムと呼ばれる期間内にスキャンサイクルを完了することが保証されています。そうでない場合は、タイムエラーイベントが生成されます。

- スキャンサイクルでは、プロセスイメージからデジタルおよびアナログの各出力の 現在値が読み取られ、自動 I/O 更新を行うように構成されている(デフォルトの構成)CPU、SB、SM の物理的な出力にその値が書き込まれます。 命令によって物理 的な出力へのアクセスが行われたときは、出力プロセスイメージと物理的な出力の 両方の更新が行われます。
- 自動 I/O 更新を行うように構成されている(デフォルトの構成)CPU、SB、SM の物理的な出力の現在の値を読み取り、それをプロセスイメージに書き込むことによって、スキャンサイクルが続行されます。 命令によって物理的な入力へのアクセスが行われたときは、その物理的な入力の値のアクセスは行われますが、入力プロセスイメージの更新は行われません。
- 入力の読み取りが終了すると、ユーザープログラムの最初の命令から最後の命令までが実行されます。 これには、すべてのプログラムサイクル OB と、それに関連するすべての FC および FB が含まれます。 プログラムサイクル OB は、OB 番号の小さなものから順に実行されます。

スキャンサイクルを通じて定期的に発生する通信処理によって、ユーザープログラムの 実行が中断されることがあります。

自己診断には、システムおよび I/O モジュールの状態の定期チェックが含まれています。 割り込みイベントは、スキャンサイクルのどこでも発生する可能性があります。 イベントが発生すると、CPU はスキャンサイクルを中断して、そのイベントを処理するように構成されている OB を呼び出します。 OB によるイベントの処理が終了すると、CPU は、ユーザープログラムの実行を割り込み発生時点から再開します。

オーガニゼーションブロック(OB)

OB は、ユーザープログラムの実行を制御します。 OB には固有の番号が必要です。 200 未満のデフォルトの OB 番号は予約されています。他の OB の番号は 200 以上でなければなりません。

CPU 内の所定のイベントによってオーガニゼーションブロックの実行が開始されます。 OB の相互呼び出しおよび FC および FB からの呼び出しを行うことはできません。 OB を開始できるのは診断割り込みや期間などの開始イベントだけです。 CPU は、OB の優先度に基づいて OB の処理を行います。優先度の高い OB は優先度の低い OB よりも先に実行されます。 優先度が最も低いのは 1 (メインプログラムサイクル用)で、優先度が最も高いのは 27 (タイムエラー割り込み用)です。

OBは、次の操作を制御します。

- プログラムサイクル OB は、CPU が RUN モードの間、周期的に実行されます。 プログラムのメインブロックはプログラムサイクル OB です。ここにプログラムを制御する命令を配置したり、ここからユーザーブロックを呼び出します。 複数のプログラムサイクル OB を使用して、番号順に実行することができます。 OB 1 がデフォルトです。 他のプログラムサイクル OB の番号は 200 番以上でなければなりません。
- スタートアップ OB は、RUN モードでの電源投入およびコマンドによる STOP モードから RUN モードへの移行を含め、CPU のモードが STOP から RUN に変更されたときに 1 回実行されます。 スタートアップ OB の実行が終了すると、引き続いて「プログラムサイクル」OB の実行が開始されます。 複数のスタートアップ OB を使用することができます。 OB 100 がデフォルトです。 デフォルト以外の OB には OB 200 以上の番号を使用する必要があります。
- 遅延 OB は、スタート割り込み(SRT_DINT)命令によるイベントの構成後、指定した 間隔をおいて実行されます。 遅延時間は、拡張命令 SRT_DINT の入力パラメータ で指定します。 遅延 OB は、指定した時間が経過したときに、プログラムの通常の サイクリック実行に割り込みを発生させます。 構成するそれぞれの遅延イベントに 対して 1 つの OB を使用し、最大 4 の遅延イベントをいつでも構成することができ ます。 遅延 OB は OB 200 以上の番号でなければなりません。
- 周期割り込み OB は、指定した間隔で実行されます。 周期割り込み OB は、2 秒間隔など、ユーザーが定義した間隔でプログラムのサイクリック実行に割り込みを発生させます。 構成するそれぞれの周期割り込みイベントに 1 つの OB を使用して、最大 4 の周期割り込みイベント構成することができます。 この OB は OB 200 以上の番号でなければなりません。

- ハードウェア割り込み OB は、内蔵デジタル入力の立ち上がりや立ち下りおよび HSC イベントなど、対応するハードウェアイベントが発生すると実行されます。 ハードウェア割り込み OB は、ハードウェアイベントからの信号に応答して、プログラムの通常のサイクリック実行に割り込みを発生させます。 イベントはハードウェア構成のプロパティで定義します。 構成するそれぞれのハードウェアイベントに対して 1 つの OB を使用することができます。 この OB は OB 200 以上の番号でなければなりません。
- タイムエラー割り込み OB は、タイムエラーが検出されたときに実行されます。タイムエラー割り込み OB は、最大サイクルタイムが経過した場合に、プログラムのサイクリック実行に割り込みを発生させます。最大サイクルタイムは、PLC のプロパティで定義します。タイムエラーイベント用としてサポートされている OB 番号は OB 80 のみです。 OB 80 が存在するときに実行する操作を、エラーを無視するか STOP モードに移行するかのどちらかに構成することができます。
- 診断エラー割り込み OB は、診断エラーが検出および報告されたときに実行されます。 診断 OB は、診断可能な(診断エラー割り込みが有効設定されている)モジュールがエラーを検出した場合に、通常のプログラムのサイクリック実行に割り込みを発生させます。 診断エラーイベント用としてサポートされている OB 番号は OB 82 のみです。 プログラム内に診断 OB が存在しない場合、このエラーを無視するかSTOP モードに移行するように、CPU を構成することができます。

3.1.2 イベント実行の優先度およびキュー

CPU の処理はイベントによって制御されます。 割り込み OB の実行開始はイベントによって決定されます。 ブロックの作成時、デバイスの構成時、または ATTACH または DETACH 命令によって、イベントに対応する割り込み OB を指定します。 プログラムサイクルイベントや周期イベントのように定期的に発生するイベントもあります。 また、スタートアップイベントや遅延イベントのように 1回しか発生しないイベントもあります。 入力ポイントでのエッジイベントや高速カウンタイベントなど、ハードウェアによってトリガされた変化に応答して発生するイベントもあります。 また、診断エラーイベントやタイムエラーイベントのように、エラーが存在するときにのみ発生するイベントもあります。 イベント割り込み OB の処理順序を決定するために、イベントのプロパティ、優先度グループ、キューが使用されます。

プログラムサイクルイベントは、各プログラムサイクル(またはスキャン)ごとに1回実行されます。 プログラムサイクルの間、CPU は出力を書き込み、入力を読み取り、プログラムサイクル OB を実行します。 プログラムサイクルイベントは必須であり、常に有効設定にしておく必要があります。 プログラムサイクル OB が存在しないことも、プログラムサイクルイベント用として選択されている OB が複数存在することもあ

ります。 プログラムサイクルイベントが発生すると、最も若い番号のプログラムサイクル OB(通常は OB1)が実行されます。 他のプログラムサイクル OB は、プログラムサイクル時間内に番号順に順次実行されます。

周期割り込みイベントを使用して、指定した間隔で割り込み OB を実行するように構成することができます。 OB を作成し、周期割り込みイベントに指定して周期イベントを構成します。周期イベントは、プログラムサイクルに割り込みを発生させ、周期割り込み OB を実行します。周期イベントはプログラムサイクルイベントよりも高い優先度グループに属しています。 周期イベントに関連付けることのできる周期割り込みイベントは1つだけです。 CPU は最大 4 つの周期割り込みイベントをサポートしています。 周期割り込み OB には、位相をシフトすることによって、同じ実行間隔の周期割り込みをずらして実行することができるように、位相シフトプロパティが存在します。

スタートアップイベントは、STOP モードから RUN モードへの移行時に 1 回発生し、スタートアップ OB が実行されます。 スタートアップイベントに応答する OB を複数 選択することができます。 スタートアップ OB は番号順に実行されます。

遅延イベントを使用して、指定した時間の経過後に割り込み OB を実行するように構成することができます。 遅延時間は SRT_DINT 命令を使用して指定します。 遅延イベントは、プログラムサイクルに割り込みを発生させて、遅延割り込みイベントを実行します。 遅延イベントに割り付けることのできる遅延割り込み OB は 1 つだけです。 CPU は最大 4 の遅延イベントをサポートしています。

ハードウェア割り込みイベントは、入力ポイントの立ち上がりや立ち下りおよび HSC (高速カウンタ)イベントなど、ハードウェア内の変化によってトリガされます。 それぞれのハードウェア割り込みイベントに対して 1 つの割り込み OB を割り付けることができます。 ハードウェアイベントはデバイスの構成で有効設定にします。 この OB は、デバイスの構成またはユーザープログラム内で ATTACH 命令を使用して、イベントに割り付けます。 CPU は複数のハードウェア割り込みイベントをサポートしています。正確なサポート可能なハードウェア割り込み OB の数は、CPU モデルおよび入力ポイントの数に依存します。

タイムエラーおよび診断エラー割り込みイベントは、CPU がエラーを検出するとトリガされます。 これらのイベントは、他の割り込みイベントよりも高い優先度グループに属し、遅延、周期、ハードウェアの各割り込みイベントの実行を中断することができます。 タイムエラー割り込みイベントおよび診断エラー割り込みイベントのそれぞれに 1 つの OB を割り付けることができます。

イベント実行の優先度およびキューを理解する

キュー内に保留するイベントの数は、イベントのタイプによって異なるキューを使用することによって制限されます。 所定のイベントタイプの保留イベントの数が限度に達すると、その次のイベントは失われます。 キューのオーバーフローの詳細については、次の「タイムエラーイベントを理解する」のセクションを参照してください。

各 CPU イベントには優先度が割り付けられていて、イベントの優先度は優先度グループに分類されています。 キューの深さ、優先度グループ、サポートされている CPU イベントの優先度を下の表にまとめてあります。

注記

優先度または優先度グループの割り付けおよびキューの深さは変更できません。

一般に、イベントの実行は優先度順に(優先度の高いものから先に)行われます。 イベントの優先度が同じ場合は、先に発生したものから順にサービスが行われます。

イベントタイプ (OB)	数量	有効な OB 番 号	キュ ーの 深さ	優先度 グルー プ	優先度
プログラムサイ クル	プログラムサイクルイベ ント ×1 複数の OB 可	1 (デフォル ト) 200 番以上	1	1	1
スタートアップ	スタートアップイベント ×1 ¹ 複数の OB 可	100 (デフォル ト) 200 番以上	1		1
遅延	遅延イベント×4 イベントあたりの OB×1	200 番以上	8	2	3
周期	周期イベント×4 イベントあたりの OB×1	200 番以上	8		4
エッジ	立ち上がりエッジイベン ト×16 立ち下がりエッジイベン ト×16 イベントあたりの OB×1	200番以上	32		5

イベントタイプ (OB)	数量	有効な OB 番 号	キュ ーの 深さ	優先度 グルー プ	優先度
HSC	CV = PV イベント×6 方向変化イベント×6 外部リセットイベント×6 イベントあたりの OB×1	200 番以上	16		6
診断エラー	1イベント	82番のみ	8		9
	タイムエラーイベント ×1 最大サイクルタイムイベ ント ×1	80 番のみ	8	3	26
2xMaxCycle タ イムイベント	2xMaxCycle タイムイベン ト×1	OB 呼び出し なし	-	3	27

1スタートアップイベントの特殊事例

- スタートアップイベントの実行が完全に終了してからプログラムサイクルイベントが開始されるため(オペレーティングシステムによって制御)、スタートアップイベントとプログラムサイクルイベントが同時に発生することはありません。
- スタートアップイベントを中断することができるイベントはありません。 スタートアップイベントの間に発生したイベントはキューに格納され、スタートアップイベントが終了した後で処理されます。

OB の実行が開始された後に、同じ優先度グループ以下の他のイベントの発生によって、その OB の処理を中断することはできません。 現在の OB が完了できるように、このようなイベントはキューに格納され、後で処理されます。

ただし、高い優先度グループのイベントの場合は、現在の OB が中断され、CPU は高い優先度のイベントを先に実行します。 高い優先度のイベントの実行が終了した後で、CPU は、グループ内の優先度に基づいて、この高い優先度グループのキューに格納されている他のイベントを実行します。 この高い優先度グループ内に(キューに格納されて)保留されているイベントがなくなると、CPU は、元の低い優先度のグループに戻って、待たせておいた OB の処理をその OB の処理を中断した時点から再開します。

割り込み待ち時間

割り込みイベント待ち時間(イベントが発生したとの CPU の通知から、そのイベントのサービスを行う OB 内の最初の命令の実行を CPU が開始するまでの時間)は、プログラムサイクル OB が、この割り込みイベントの発生時に唯一のイベントであるとすれば 210 µsec です。

タイムエラーイベントを理解する

いくつかのタイムエラー条件が発生すると、タイムエラーイベントがトリガされます。 下記のタイムエラーがサポートされています。

- 最大サイクルタイムを超える
- 要求された OB 開始不可
- キューのオーバーフローが発生

最大サイクルタイム経過条件は、設定した最大スキャンサイクル時間内にプログラムサイクルが完了しなかった場合に発生します。 最大サイクルタイム経過条件、最大スキャンサイクルタイムの構成方法、およびサイクルタイマのリセット方法の詳細については、「サイクルタイムを監視する (ページ 51)」を参照してください。

要求された OB 開始不可条件は、周期割り込みまたは遅延割り込みによって要求された OB がすでに実行中である場合に発生します。

キューのオーバーフローの発生条件は、割り込みの処理能力を超えて割り込みが発生している場合に成立します。 (キュー内の)保留イベントの数は、イベントのタイプによって異なるキューを使用して制限されます。 イベントが発生したときに、そのイベントに割り付けられているキューに空きがないと、タイムエラーイベントが生成されます。

すべてのタイムエラーイベントは、OB 80 が存在する場合は、その実行をトリガします。OB 80 が存在しない場合は、CPUはこのエラーを無視します。 サイクルタイマをリセットせずに、同一のプログラムサイクル内で 2 つの最大サイクルタイム経過条件が発生した場合、OB 80 が存在するかどうかに関係なく、CPUはSTOPモードに移行します。詳細については、「サイクルタイムを監視する」 (ページ 51)のセクションを参照してください。

OB 80 には、どのイベントおよび OB がタイムエラーを生成したかを決定するのに役立 つスタートアップ情報が含まれています。 これらのスタートアップ値を調べて対応す る操作を実行する命令を OB 80 内にプログラムすることができます。 OB 80 では、次のスタートアップロケーションがサポートされています。

入力	データタイプ	説明
fault_id	BYTE	16#01 - 最大サイクルタイム経過
		16#02 - 要求された OB が開始不可
		16#07 および 16#09 - キューのオーバーフローが発生
csg_OBnr	OB_ANY	エラー発生時に実行されていた OB の番号
csg_prio	UINT	エラーを起こした OB の優先度

新しいプロジェクトの作成時にはタイムエラー割り込み OB 80 は存在しません。 必要 に応じて、ツリー内の[Program blocks]の下の[Add new block]をダブルクリックし、[Organization block]を選択して[Time error interrupt]を選択し、作成したプロジェクトに タイムエラー割り込み OB 80 を追加します。

診断エラーイベントを理解する

デバイスによっては診断エラーを検出して報告する機能を備えたものがあります。 複数の診断エラー条件のいずれかの発生または削除によって、診断エラーイベントがトリガされます。 下記の診断エラーがサポートされています。

- ユーザー電源なし
- 上限超過
- 下限超過
- 断線
- 短絡

すべての診断エラーイベントは、OB 82 が存在する場合は OB 82 の実行をトリガします。 OB 82 が存在しない場合は、CPU はこのエラーを無視します。 新しいプロジェクトの作成時には診断エラー割り込み OB 82 は存在しません。 必要に応じて、ツリー内の[Program blocks]の下の[Add new block]をダブルクリックし、[Organization block]を選択して[Diagnostic error interrupt]を選択し、作成したプロジェクトに診断エラー割り込み OB 82 を追加します。

OB 82 には、このイベントがエラーの発生または削除のどちらによってトリガされたのかの決定、およびエラーを報告しているデバイスおよびチャンネルの決定に役立つスタートアップ情報が含まれています。 これらのスタートアップ値を調べて対応する操作を実行する命令を OB 82 内にプログラムすることができます。 OB 82 では、次のスタートアップロケーションがサポートされています。

入力	データタイプ	説明
IOstate	WORD	デバイスの I/O の状態
laddr	HW_ANY	エラーを報告しているデバイスまたはユ ニットのハードウェア ID
channel	UINT	チャンネル番号
multierror	BOOL	複数のエラーが存在する場合は 1 (<i>初期</i> のバージョンではサポートされていません)

IO_state の 4 ビットは、このイベントがエラーの発生または削除のどちらによってトリガされたのかを表しています。 この 4 ビットでは、エラーが存在する(断線など)場合は 1、エラーが存在しなくなった場合は 0 です。

ラダー入力には、エラーを返したデバイスまたは機能ユニットのハードウェア ID (HW ID)が含まれています。 HW ID は、デバイスビューまたはネットワークビューにコンポーネントを挿入すると自動的に割り付けられ、PLC タグの[Constants]タブに表示されます。 HW ID の名前も自動的に割り付けられます。 PLC タグの[Constants]タブにあるこれらのエントリを変更することはできません。

チャンネル番号は入力ポイント(アナログまたはデジタル)の場合は 0 から始まり、出力ポイント(アナログまたはデジタル)の場合は 64 から始まります。 デバイスに入力と出力の両方が含まれている場合は、入力と出力を識別するためのオフセットが必要です。 ユーザー電源なしなど、エラーがデバイスまたは機能ユニット全体に影響する場合は、チャンネル番号ワードの最上位ビットがセットされます(チャンネル番号 32768)。

サイクルタイムを監視する

サイクルタイムは、CPU オペレーティングシステムが RUN モードの周期部分を実行するの に必要な時間です。 CPU にはサイクルタイムを監視するメソッドが 2 種類用意されています。

- 最大スキャンサイクルタイム
- 最小固定スキャンサイクルタイム

スキャンサイクルの監視は、スタートアップイベントが完了した後に開始されます。 この機能の構成は、CPU の[Device Configuration]の下の[Cycle time]の下に表示されます。

CPU は常にスキャンサイクルの監視を行い、最大スキャンサイクルタイムが経過した場合は、それに対応します。 構成されている最大スキャンサイクルタイムが経過した場合は、エラーが生成され、次のどちらかの方法で処理されます。

- タイムエラー割り込み OB 80 が存在しない場合、CPU はエラーを生成し、ユーザープログラムの実行を続けます。
- タイムエラー割り込み OB 80 が存在する場合、CPU は OB 80 を実行します。

RE_TRIGR 命令(サイクルタイム監視の再トリガ)を使用して、サイクルタイムを計時しているタイマをリセットすることができます。 ただし、この命令が機能するのはプログラムサイクル OB 内で実行された場合だけです。RE_TRIGR 命令を OB 80 内で実行した場合は無視されます。同じプログラムサイクル内で、最大スキャンサイクルタイムを 2 回超過した場合は(その 2 回の間に RE_TRIGR 命令を使用せずに)、CPU は直ちに STOP モードに移行します。 RE_TRIGR 命令の反復実行を使用して、エンドレスループまたは非常に長時間のスキャンを作成することができます。

通常は、スキャンサイクルはできるだけ短い時間内に実行し、現在のスキャンサイクルが完了すると、すぐに次のスキャンサイクルを開始します。 ユーザープログラムおよび通信タスクによって、スキャンサイクルの期間はスキャンごとに異なることがあります。 この変動を除去するために、CPU は、オプションの最小固定スキャンサイクルタイム(固定スキャンサイクル)をサポートしています。 このオプションの設定を有効にし固定スキャンサイクルタイムを ms で指定した場合、CPU は、各 CPU スキャンの完了に必要な最小スキャンサイクルタイムを±1 ms の精度に維持します。

指定した最小スキャンサイクルタイムよりも短い時間で通常のスキャンサイクルが完了した場合、CPU は実行時診断や通信要求の処理の実行に、スキャンサイクルタイムの残りの時間を使用します。 このようにして、CPU は常にスキャンサイクルの完了のために固定長の時間を使用します。

指定した最小スキャンサイクルタイムでスキャンサイクルが完了しなかった場合、CPU は 通常のスキャンを完了し(通信処理を含む)、最小スキャンタイムを超過したことに対するシステム応答を作成しません。 サイクルタイムの監視機能の範囲とデフォルト値を次の表に 示します。

サイクルタイム	範囲(ms)	デフォルト
最大スキャンサイクルタイム1	1~6000	150 ms
最小固定スキャンサイクルタイム	1~最大スキャンサイクルタイム	Disabled
2		

- 1 最大スキャンサイクルタイムは常に有効設定になっています。 1~6000 ms の範囲でサイクルタイムを設定します。デフォルト値は 150 ms です。
- ² 最小固定スキャンサイクルタイムはオプションで、デフォルトでは無効設定になっています。 必要に応じて、1 ms~最大スキャンサイクルタイムの範囲のサイクルタイムを設定します。

サイクルタイムと通信負荷を設定する

[Device configuration]の CPU プロパティを使用して、下記のパラメータを設定することができます。

• Cycle time (サイクルタイム): 最大スキャンサイクルタイムを入力することができます。 また、最小固定スキャンサイクルタイムを入力することもできます。



• Communications load (通信負荷): 通信タスクに割り付ける時間の割合を設定することができます。



詳細については、「サイクルタイムを監視する」(ページ51)を参照してください。

3.1.3 **CPU**メモリ

メモリ管理

CPU には、ユーザープログラム、データ、および構成を保存するための、次のメモリ 領域が提供されています。

- ロードメモリは、ユーザープログラム、データ、および構成を保存するための不揮発性メモリ領域です。 プロジェクトを CPU にダウンロードすると、まずこのロードメモリ領域に保存されます。 この領域は、メモリカード(存在する場合)または CPU のどちらかに配置されています。 この不揮発性メモリ領域の内容は、電源が切断された場合にも保持されます。 メモリカードは、CPU 内蔵のメモリよりも大きな格納空間をサポートします。
- ワークメモリは、ユーザープログラムの実行中に、ユーザープロジェクトのいくつかの要素を格納しておくための揮発性メモリ領域です。 CPU は、プロジェクトの必要な要素をロードメモリからワークメモリにコピーします。 このワークメモリは揮発性メモリ領域であるので、電源が切断されると情報が失われ、電源が復旧すると CPU によって復元されます。
- 保持メモリは、ワークメモリの値の一部を格納しておくための不揮発性メモリ領域です。 保持メモリ領域は、電源が切断された時でも、選択したユーザーメモリの情報を保持するために使用されます。 電源断が発生しても、CPU は、指定した限られたロケーションの値を保持することができます。 電源が復旧すると、保持されていた値が復元されます。

現在のプロジェクトのメモリ使用状況を表示するには、CPU (または、そのブロックの1つ)を右クリックし、コンテキストメニューの[Resources]を選択します。 現在の CPU のメモリの使用状況を表示するには、[Online and diagnostics]をダブルクリックし、[Diagnostics]を展開して[Memory]を選択します。

保持メモリ

電源障害によるデータの消失を防ぐには、データを保持するようにマークすることで、そのデータの消失を防ぐことができます。 下記のデータを保持するように設定することができます。

- ビットメモリ(M): PLC タグテーブルまたは割り付けリストで、保持するビットメモリの正確な幅を定義することができます。 保持ビットメモリは常に MB0 から始まり、指定したバイト数だけ連続しています。 PLC タグテーブルまたは割り付けリストの[Retain]ツールバーアイコンをクリックすることによって、この値を指定することができます。 MB0 から始まる M バイト数を入力します。
- ファンクションブロックのタグ(FB): FB の作成時に[Symbolic access only]ボックスをオンにした場合、この FB のインターフェースエディタには[Retain]列が表示されます。 この列で、各タグ個別に[Retain]または[Non-Retain]のどちらかを選択することができます。 この FB をプログラムエディタに配置したときに作成されるインスタンス DB にも、この保持列が表示されますが、この列は表示専用であり、「シンボリックアクセスのみ」に構成されている FB 用のインスタンス DB インターフェースエディタ内で保持ステータスを変更することはできません。

FB の作成時に[Symbolic access only]ボックスをオフにした場合、この FB のインターフェースエディタには[Retain]列は表示されません。 この FB をプログラムエディタに挿入したときに作成されるインスタンス DB には、編集に使用可能な[Retain]列が表示されます。 この場合、任意のタグの[Retain]オプションを選択すると、すべてのタグが選択されます。 同様に、任意のタグの[Retain]オプションの選択を解除すると、すべてのタグの選択が解除されます。 「シンボリックアクセスのみ」に構成されていない FB の場合は、インスタンス DB エディタで保持ステータスを変更することができますが、すべてのタグが同じ保持ステータスに設定されます。

FB を作成した後で、「シンボリックアクセスのみ」のオプションを変更することはできません。このオプションを選択することができるのは FB の作成時のみです。既存のタグが「シンボリックアクセスのみ」に構成されているかどうかを調べるには、プロジェクトツリーで FB を右クリックし、[Properties]を選択して、[Attributes]を選択します。

- グローバルデータブロックのタグ グローバル DB での保持ステータスの割り付けについては、FB の場合と同様です。 シンボリックアドレス指定の設定に基づいて、グローバルデータブロックの個別のタグまたはすべてのタグのどちらかに対して保持ステータスを定義することができます。
 - DB の[Symbolic access only]属性をオンにした場合は、個別のタグに対して保持ステータスを設定することができます。
 - DB の[Symbolic access only]属性をオフにした場合、この保持ステータスの設定 は DB のすべてのタグに適用され、すべてのタグを保持するかタグをまったく保持しないかのどちらかになります。

合計 2048 バイトのデータを保持することができます。 使用可能な容量を調べるには、PLC タグテーブルまたは割り付けリストで、[Retain]ツールバーアイコンをクリックします。 M メモリの保持領域の範囲を指定するフィールドですが、M および DB 用を組み合わせた使用可能な保持メモリの総残量が 2 行目に表示されます。

診断バッファ

CPU は、それぞれの診断イベントのエントリを格納する診断バッファをサポートしています。 各エントリには、イベントの発生日時、イベントのカテゴリ、イベントの説明が含まれています。 エントリは、最も新しく発生したものを一番上にして、発生した日時順に表示されます。 CPU に電源が接続されている間、このログには最大過去 50のイベントが表示されます。 ログの容量に空きがなくなると、最も古いものが消去され、新しいものが追加されます。 電源が切断されても、過去 10のイベントが保持されます。

診断バッファには、次のタイプのイベントが記録されます。

- システム診断イベント(CPU エラーやモジュールエラーなど)
- CPU のモードの変化(電源投入、STOP モードへの移行、RUN モードへの移行)

診断バッファにアクセスするには、オンラインになっている必要があります。 Online diagnostics / Diagnostics / Diagnostics buffer の下のログを開きます。 トラブルシューティングおよびデバッグの詳細については、「オンラインおよび診断」の章を参照してください。

時計

CPU は時計をサポートしています。 CPU の電源が切断されている間も、このクロックの実行に必要なエネルギーがスーパーコンデンサから供給されます。 CPU に電源が接続されているときにスーパーコンデンサへの充電が行われます。 CPU の電源投入時間が 2 時間を超えると、スーパーコンデンサには 10 日間のクロックの実行に必要な充電が行われます。

時計は、システム時刻(協定世界時(UTC))にセットされています。 STEP 7 Basic では、時計がシステム時刻にセットされています。 システム時刻またはローカル時刻を読み取るための命令が用意されています(システム時刻用 RD_SYS_T およびローカル時刻用 RD_LOC_T)。 ローカル時刻は、[CPU Clock]デバイス構成で設定した時間帯とサマータイムオフセットを使用して計算されます。

[Time of day]プロパティで、CPU の時計を設定します。 サマータイムも使用することができます。 サマータイムの開始時刻と終了時刻を指定します。 時計をセットするには、オンラインおよび CPU の[Online diagnostics]表示にする必要があります。 [Set time of day]オプションを使用します。

システムメモリとクロックメモリ

「システムメモリ」および「クロックメモリ」用のバイトを有効設定にするには CPU のプロパティを使用します。 プログラム内ロジックで、これらのファンクションの個別のビットを参照することができます。

- システムメモリとして M メモリに 1 バイトを割り付けることができます。 このシステムメモリのバイトの次の 4 ビットをユーザープログラムから参照することができます。
 - 「Always 0 (low)」 指定されたビットは常に 0 にセットされています。
 - 「Always 1 (high)」 指定されたビットは常に 1 にセットされています。
 - 「Diagnostic graph changed」(診断グラフ変更)ビット CPU が診断イベントを ログに記録してから 1 スキャンの間、指定したビットは 1 にセットされています。 プログラムサイクル OB の最初の実行が終了するまで CPU は「Diagnostic graph changed」ビットをセットしないため、ユーザープログラムで診断変更がスター トアップ OB の実行時に発生したのか、プログラムサイクル OB の最初の実行で 発生したのかを検出することができません。
 - 「First scan」(最初のスキャン)ビット スタートアップ OB の完了後の最初のスキャンの間はこのビットに 1 がセットされます。 (最初のスキャンの実行後に「First scan」ビットは 0 にセットされます)。
- クロックメモリとして M メモリに 1 バイトを割り付けることができます。 クロックメモリとしてのバイトの各ビットは、方形波パルスを生成します。 クロックメモリのバイトは、0.5 Hz (低速)~10 Hz (高速)の 8 種類の周波数を供給します。 これらのビットを制御ビットとして使用し、特にエッジ命令と組み合わせて、ユーザープログラム内の操作を周期的にトリガすることができます。

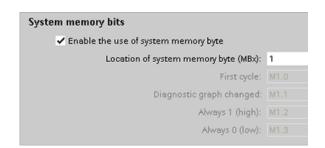
これらのバイトは、CPUのSTOPモードからSTARTUPモードへの移行時に初期化されます。 クロックメモリの各ビットは、STARTUPモードとRUNモードの期間を通して同期して変化します。

/ 注意

システムメモリまたはクロックメモリのビットを上書きすると、これらのファンクションのデータが破損し、ユーザープログラムの誤動作が発生し、人的傷害や物的損害の原因になることがあります。

システムメモリもクロックメモリも M メモリ上の予約されている領域ではないため、 命令または通信によってこれらのロケーションへの書き込みが発生し、データが破壊 される可能性があります。

これらのファンクションの正しい動作を保証するために、これらのロケーションへの データの書き込みを防止する必要があります。また、プロセスまたは機器には緊急停 止回路を必ず実装してください。



システムメモリは、次の条件に対してオンになる(値=1)1 バイトを構成します。

- 「First scan」(最初のスキャン): RUN モードでの最初のスキャン サイクルの間オンになります。
- 「Diagnostic graph changed」(診 断グラフ変更)
- 「Always 1 (high)」(常に 1 (高)): 常にオンです
- 「Always 0 (low)」(常に 0 (低)): 常にオフです

クロックメモリは、一定の周期で個別のビットがオンとオフを繰り返す 1バイトを構成します。

それぞれのクロックフラグは、Mメモリの対応するビット上に方形波パルスを生成します。 これらのビットを制御ビットとして使用し、特にエッジ命令と組み合わせて、ユーザーコード内の操作を周期的にトリガすることができます。

CPU が STOP モードになるときの出力値の状態を設定する

CPU が STOP モードになるときの、デジタルおよびアナログ出力の状態を設定することができます。 CPU、SB、または SM の任意の出力に対して、出力の値を保持するのか、代替値を使用するのかを設定することができます。

• 「Substituting a specified output value (default)」(指定した出力値に代替する(デフォルト)): その CPU、SB、または SM デバイスの各出力(チャンネル)に対する代替値を入力します。

デジタル出力チャンネルのデフォルトの代替値は **OFF** で、アナログ出力チャンネルのデフォルトの代替値は **O** です。

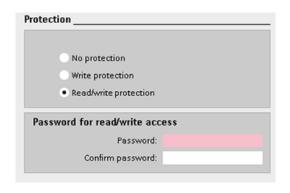
• 「Freezing the outputs to remain in last state」(出力を最終の状態で保持する): 出力は、RUN モードから STOP モードへの移行時の現在の値を保持します。 電源投入後、出力はデフォルトの代替値にセットされます。

[Device Configuration]で、出力の状態を構成します。 個別のデバイスを選択し、 [Properties]タブを使用して、各デバイスの出力を設定します。

CPU が RUN モードから STOP モードに移行するとき、CPU はプロセスイメージを保持し、構成に基づいてデジタル出力およびアナログ出力の両方に対して適切な値を書き込みます。

3.1.4 S7-1200 CPUのパスワード保護

特定のファンクションへのアクセスを制限するために、CPU には 3 レベルのセキュリティが備わっています。 CPU に対するセキュリティレベルおよびパスワードを設定して、パスワードを使用せずにアクセスできるファンクションとメモリ領域を制限します。



パスワードを構成するには、次の手順を 実行します。

- 1. [Device configuration]で CPU を選択します。
- 2. インスペクタウインドウの [Properties]タブを選択します。
- 3. [Protection]プロパティを選択し、保護レベルを選択して、パスワードを入力します。

パスワードには大文字と小文字の区別が あります。

各レベルに、パスワードを使用せずにアクセスできるファンクションがあります。 CPU のデフォルトの条件は無制限で、パスワード保護のない状態です。 CPU へのアクセスを制限するには、CPU のプロパティを設定して、パスワードを入力します。

ネットワークを通じてのパスワードの入力は、CPU のパスワード保護情報を漏洩しません。 パスワード保護された CPU には同時に一人のユーザーしか無制限アクセスを行うことができません。 パスワード保護は、通信ファンクションを含め、ユーザープログラムの実行には適用されません。 正しいパスワードを入力することによって、すべてのファンクションにアクセスすることができます。

PLC 同士の通信(プログラムブロック内の通信命令を使用)は、CPU 内のセキュリティレベルによる制限を受けません。 HMI 機能も制限されません。

セキュリティレベ ル	アクセス制限
保護なし	パスワード保護を行わずに、すべてのファンクションにアクセ スすることができます。
書き込み保護	パスワードを使用せずに、HMI と PLC 間通信の情報へのアクセスが許可されます。
	CPU の修正(CPU への書き込み)および CPU のモード (RUN/STOP)の変更にはパスワードが必要です。
読み取り/書き込 み保護	パスワードを使用せずに、HMI と PLC 間通信の情報へのアクセスが許可されます。
	CPU 内のデータの読み取り、CPU の修正(CPU への書き込み)および CPU のモード(RUN/STOP)の変更にはパスワードが必要です。

3.1.5 パスワードを忘れてしまった場合の復元

パスワード保護された CPU のパスワードを忘れてしまった場合、空のトランスファーカードを使用して、パスワード保護されたプログラムを削除します。 空のトランスファーカードは、CPU の内部ロードメモリを消去します。 その後で、STEP 7 Basic から CPU に新しいユーザープログラムをダウンロードします。

空のトランスファーカード作成および使用の詳細については、トランスファーカード (ページ 83)のセクションを参照してください。

! 警告

実行中の CPU にトランスファーカードを挿入すると、CPU は STOP モードに移行します。 制御装置が安全でない状態に陥り、被制御機器に予測できない動作が発生することがあります。 このような予測できない動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

CPU を RUN モードに設定する前に、トランスファーカードを抜き取る必要があります。

3.2 データストレージ、メモリ領域、アドレス指定

CPU には、ユーザープログラム実行中のデータを保存するためのオプションがいくつか用意されています。

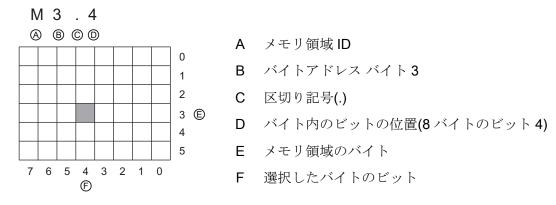
- グローバルメモリ: CPU には、入力(I)、出力(Q)、ビットメモリ(M)など、さまざまな専用メモリ領域が用意されています。 このメモリには、プログラム内のどこからでもアクセスできます。
- データブロック(DB): プログラムで使用するデータを保存するために、ユーザープログラムに DB を作成できます。 保存されたデータは、関連付けられているプログラムが終了しても残ります。 「グローバル」 DB にはプログラム内で使用するすべてのデータを保存することができ、インスタンス DB には対応する FB のデータだけを保存し、FB 用のパラメータで構造化します。
- テンポラリメモリ: プログラムブロックが呼び出されると、CPU のオペレーティングシステムは、ブロックの実行中に使用するテンポラリメモリまたはローカルメモリ(L)を割り付けます。 このプログラムブロックの実行が終了すると、CPU は、他のプログラムブロックの実行用にローカルメモリを割り付け直します。

それぞれのメモリロケーションには、固有のアドレスが存在します。 ユーザープログラムでは、これらのアドレスを使用して、メモリロケーション内の情報にアクセスします。

メモリ領域	説明	強制	保持
I	スキャンサイクル開始時に物理的な	なし	なし
プロセスイメージ入	入力からコピーされる		
力	CPU、SB、SM 上の物理的な入力ポ	あり	なし
I_:P	イントの即時読み取り		
(物理的な入力)			
Q	スキャンサイクル開始時に物理的な	なし	なし
プロセスイメージ出	出力からコピーされる		
力	CPU、SB、SM 上の物理的な出力ポ	あり	なし
Q_:P	イントへの即時書き込み		
(物理的な出力)			
M	制御およびデータメモリ	なし	あり
ビットメモリ			
L	ブロック用のテンポラリデータ、1つ	なし	なし
テンポラリメモリ	のブロック専用		
DB	データメモリおよび FB のパラメータ	なし	あり
データブロック	用メモリ		

3.2 データストレージ、メモリ領域、アドレス指定

それぞれのメモリロケーションには、固有のアドレスが存在します。 ユーザープログラムでは、これらのアドレスを使用して、メモリロケーション内の情報にアクセスします。 下の図に各ビットへのアクセス方法(「バイト.ビット」アドレス指定と呼ばれる)を示します。 この例では、メモリ領域の指定に続けてバイトアドレス(I=入力、3=バイト3)とビットアドレス(ビット4)をピリオド(.)で連結して指定しています。



メモリ領域(I、Q、M、DB、L)内のデータには、通常は、「バイトアドレス」フォーマットを使用して、バイト、ワード、ダブルワードとしてアクセスすることができます。メモリ内のデータのバイト、ワード、ダブルワードにアクセスするには、ビットのアドレス指定と同様の方法でアドレスを指定する必要があります。 領域 ID およびデータサイズ指定子、バイト、ワード、ダブルワードの値の開始アドレスを指定します。 サイズ指定子は B (バイト)、W (ワード)、D (ダブルワード)です(IBO、MW2O、QD8 など)。 IO.3 や Q1.7 などの参照はプロセスイメージにアクセスします。 物理的な入力または物理的な出力に即時アクセスするには、参照先に「:P」を付加します(IO.3:P、Q1.7:P、Stop:P など)。

CPU のメモリ領域内のデータにアクセスする

STEP 7 Basic では、簡単にシンボリックプログラミングを行うことができます。 通常は、PLC タグ、データブロック、または OB、FC、FB 上部にあるインターフェース内にタグを作成します。 これらのタグには、名前、データタイプ、アドレス、コメントが含まれます。 また、データブロック内に初期値を指定することもできます。 プログラミングで、命令内のパラメータにタグ名を入力することによってタグを使用することができます。 命令内のパラメータに絶対アドレス(メモリ、領域、サイズ、オフセット)を入力することもできます。 以降のセクションの例に絶対アドレスの入力方法を示します。 プログラムエディタによって、絶対アドレスの前に識別子として%が自動的に挿入されます。 プログラムエディタでの表示は「シンボリック表示」、「シンボリック表示と絶対アドレス表示」、「絶対アドレス表示」があり、切り替えることができます。

I(プロセスイメージ入力): CPU は、各スキャンサイクルのサイクリック OB の実行の前に(物理的な)入力ポイントのサンプリングを行い、取得した値を入力プロセスイメージに書き込みます。 この入力プロセスイメージをビット、バイト、ワード、ダブルワードとしてアクセスすることができます。 読み取りと書き込みの両方のアクセスが可能ですが、通常は、プロセスイメージ入力は読み取りのみを行います。

Bit	[バイトアドレス].[ビットアド レス]	10.1
バイト、ワード、ダブルワ ード	[サイズ][スタートバイトアド レス]	IB4、IW5、ID12 など

アドレスに「:P」を付加することによって、CPU、SB、または SM のデジタルおよび アナログ入力を直接読み取ることができます。 I ではなく I:P を使用してアクセスした 場合は、入力プロセスイメージではなく、アクセスしたポイントから直接データの読み 取りが行われます。 この I:P アクセスは、入力プロセスイメージの前回の更新時に作成されたコピーからではなく、ソースから直ちにデータが取得されるため、「即時読み 取り」アクセスと呼ばれます。

物理的な入力ポイントが接続先のフィールドデバイスから直接値を受信するため、これらのポイントへの書き込みは禁止されます。 つまり、I アクセスが読み取りと書き込みの両方が可能なのに対して、I:P アクセスは読み取り専用です。

3.2 データストレージ、メモリ領域、アドレス指定

また、I_:P アクセスは、単一の CPU、SB、または SM によってサポートされている入力のサイズに制限され、最も近いバイトに切り上げられます。 たとえば、2 DI / 2 DQ SB の入力が I4.0 から始まるように構成されている場合、この入力ポイントを I4.0:P および I4.1:P として、または IB4:P としてアクセスすることができます。 I4.2:P から I4.7:P までのアクセスも拒否されませんが、これらのポイントは使用されていないため無意味です。 IW4:P および ID4:P などのアクセスは、この SB に割り付けられているバイトオフセットを超えるため禁止されます。

L:P を使用してアクセスした場合、入力プロセスイメージに格納されている対応する値は影響を受けません。

Bit	[バイトアドレス].[ビットアド	10.1:P など
	レス]:P	
バイト、ワード、ダブルワ	[サイズ][スタートバイトアド	IB4:P、IW5:P、ID12:P
ード	レス]:P	など

Q (プロセスイメージ出力): CPU は、出力プロセスイメージに格納されている値を物理的な出力ポイントにコピーします。 この出力プロセスイメージをビット、バイト、ワード、ダブルワードとしてアクセスすることができます。 プロセスイメージ出力の場合は、読み取りアクセスと書き込みアクセスの両方が可能です。

Bit	Q[バイトアドレス].[ビットア ドレス]	Q1.1 など
バイト、ワード、ダブルワ	Q[サイズ][スタートバイトア	QB5、QW10、QD40 な
ード	ドレス]	ど

アドレスに「:P」を付加することによって、CPU、SB、または SM のデジタルおよびアナログ出力に直ちに書き込むことができます。 Q ではなく Q:P を使用してアクセスした場合は、アクセス先のポイントと出力プロセスイメージにデータが直接書き込まれます(両方の場所に書き込まれます)。 この Q:P アクセスは、データが直ちにターゲットポイントに送られ、ターゲットポイントは出力プロセスイメージの次回の更新を待つ必要がないため、「即時書き込み」アクセスと呼ばれることもあります。

物理的な出力ポイントが接続先のフィールドデバイスを直接制御するため、これらのポイントからの読み取りは禁止されます。 つまり、 \mathbf{Q} アクセスが読み取りと書き込みの両方が可能なのに対して、 \mathbf{Q} : \mathbf{P} アクセスは書き込み専用です。

また、Q_:P アクセスは、単一の CPU、SB、SM によってサポートされている出力のサイズに制限され、最も近いバイトに切り上げられます。 たとえば、2 DI / 2 DQ SB の入力が Q4.0 から始まるように構成されている場合、この出力ポイントを Q4.0:P および Q4.1:P として、または QB4:P としてアクセスすることができます。 Q4.2:P から Q4.7:P までのアクセスも拒否されませんが、これらのポイントは使用されていないため無意味です。 QW4:P および QD4:P のアクセスは、この SB に関連付けられているバイトオフセットを超えるため禁止されます。

Q_:P を使用してアクセスした場合は、物理的な出力と、出力プロセスイメージに格納されている対応する値の両方に影響を与えます。

Bit	Q[バイトアドレス].[ビットア ドレス]:P	Q1.1:P
バイト、ワード、ダブルワ	Q[サイズ][スタートバイトア	QB5:P、QW10:P、
ード	ドレス]:P	QD40:P など

M (ビットメモリ領域): ビットメモリ領域(M メモリ)は、プログラム途中の状態または他の制御情報を保存するための制御リレーやデータの保存の両方に使用します。 ビットメモリ領域はビット、バイト、ワード、またはダブルワードとしてアクセスすることができます。 M メモリに対しては、読み取りと書き込みの両方のアクセスを行うことができます。

Bit	M[バイトアドレス].[ビットア ドレス]	M26.7 など
バイト、ワード、ダブルワ	M[サイズ][スタートバイトア	MB20、MW30、MD50
ード	ドレス]	など

Temp (テンポラリメモリ): CPU は必要に応じてテンポラリメモリを割り付けます。 プログラムブロックが開始されるか(OB の場合)または呼び出されると(FC または FB の場合)、CPU は、そのプログラムブロック用のテンポラリメモリを割り付けます。 プログラムブロック用のテンポラリメモリの割り付けに、他の OB、FC、または FB に使用したテンポラリメモリのロケーションと同じロケーションが使用されることがあります。 CPU は、テンポラリメモリの割り付け時にはメモリの初期化を行いませんから、割り付け時にテンポラリメモリに値が格納されていることがあります。

3.2 データストレージ、メモリ領域、アドレス指定

テンポラリメモリは、一つの大きな例外を除き M メモリと類似しています。 M メモリ の有効範囲は「グローバル」ですが、テンポラリメモリの有効範囲は「ローカル」です。

- Mメモリ: どの OB、FC、FB からも M メモリにアクセスできます。つまり、ユーザープログラム内のすべての要素が、このデータをグローバルに使用することができます。
- テンポラリメモリ: テンポラリメモリのアクセスは、そのテンポラリメモリのロケーションを作成または宣言した OB、FC、または FB 内からに限定されます。 そのプログラムブロックが他のプログラムブロックを呼び出した場合でも、テンポラリメモリのロケーションはローカルのままで、呼び出し先のプログラムブロックによって共有されることはありません。 たとえば、 OB が FC を呼び出した場合、呼び出された FC が呼び出した OB のテンポラリメモリにアクセスすることはできません。

CPU は、**3** 種類の **OB** の優先度グループそれぞれにテンポラリメモリ(ローカルメモリ) を割り付けます。

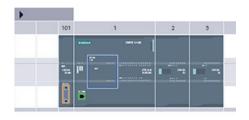
- スタートアップおよびプログラムサイクルと関連する FB および FC に 16 KB
- FB および FC を含めた標準割り込みイベントに 4 KB
- FB および FC を含めたエラー割り込みイベントに 4 KB

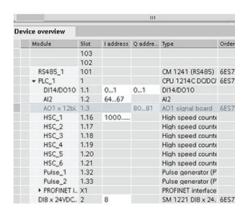
テンポラリメモリはシンボリックアドレス指定によってのみアクセスします。

DB (データブロック): DB メモリは、操作途中のステータス、FB 用の制御情報パラメータ、およびタイマやカウンタなど多くの命令に必要なデータ構造など、さまざまな種類のデータを保存するために使用します。 データブロックを読み取り/書き込みまたは読み取り専用として指定することができます。 データブロックメモリをビット、バイト、ワード、またはダブルワードでアクセスすることができます。 M メモリに対しては、読み取りと書き込みの両方のアクセスを行うことができます。 読み取り専用データブロックに対しては、読み取りのみが許可されます。

Bit	DB[データブロック番	DB1.DBX2.3 など
	号].DBX[バイトアドレス].[ビ	
	ットアドレス]	
バイト、ワード、ダブルワ	DB[データブロック番号].DB	DB1.DBB4、
ード	[サイズ][スタートバイトアド	DB10.DBW2、
	レス]	DB20.DBD8 など

CPU 内の I/O および I/O モジュールのアドレス指定





構成画面に CPU および I/O モジュールを追加 すると、I アドレスと Q アドレスが自動的に割 り付けられます。

デフォルトのアドレス指定を変更するには、構成画面のアドレスフィールドを選択して新しい番号を入力します。 デジタルの入力および出力には、8 ビットバイトで割り付けられます。 そのモジュールがすべてのポイントを使用するかどうかは関係ありません。 アナログの入力および出力には、2 ポイント (4 バイト) のグループで割り付けられます。 この例では、

DI16 のアドレスを 8..9 ではなく 2..3 に変更できました。誤ったサイズや他のアドレスと競合するアドレス範囲を入力しないようにツールが支援してくれます。

2つの **SM** を装着した **CPU 1214C** の例を図に示します。

3.3 データタイプ

データタイプは、データの要素と、そのデータの解釈方法の両方を指定するために使用します。 命令パラメータのそれぞれは少なくとも 1 つのデータタイプをサポートしています。 複数のデータタイプをサポートしているパラメータもあります。 命令のパラメータフィールドの上にカーソルを移動すると、そのパラメータがサポートしているデータタイプが表示されます。

公式のパラメータは命令の ID で、その命令が使用するデータのロケーションを表します (ADD 命令の IN1 入力など)。 実際のパラメータはメモリのアドレスまたは、命令が使用するデータを格納した定数です(%MD400 "Number_of_Widgets"など)。 指定した 実際のパラメータのデータタイプが、命令によって指定された公式のパラメータがサポートしているデータタイプと一致している必要があります。

実際のパラメータを指定するときに、タグ(シンボル)またはメモリの絶対アドレスのどちらかを指定する必要があります。 タグによって、シンボリック名(タグ名)とデータタイプ、メモリ領域、メモリのアドレス、コメントが関連付けられます。PLC タグエディタまたはブロック(OB、FC、FB、DB)用のインターフェースエディタでタグを作成することができます。 関連付けられているタグが存在しない絶対アドレスを入力する場合は、サポートされているデータタイプに適合するサイズを使用する必要があります。 絶対アドレスを入力するとデフォルトのタグが作成されます。

また、入力パラメータの多くに、定数値を入力することができます。 下の表に、定数入力の例を含めて、サポートされている基本的なデータタイプを示します。 String を除き、すべてのデータタイプを PLC タグエディタおよびブロックインターフェースエディタの両方で使用することができます。 String は、ブロックインターフェースエディタでのみ使用することができます。 下の表に基本的なデータタイプの定義を示します。

データタ イプ	サイズ(ビ ット)	範囲	定数入力の例
Bool	1	0~1	TRUE, FALSE, 0, 1
Byte	8	16#00∼16#FF	16#12, 16#AB
Word	16	16#0000~16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000~16#FFFFFFF	16#02468ACE
Char	8	16#00∼16#FF	'A', 't', '@'
Sint	8	-128~127	123, -123
Int	16	-32,768~32,767	123, -123
Dint	32	-2,147,483,648~2,147,483,647	123, -123

データタ イプ	サイズ(ビ ット)	範囲	定数入力の例
USInt	8	0~255	123
UInt	16	0~65,535	123
UDInt	32	0~4,294,967,295	123
Real	32	+/-1.18 x 10 ⁻³⁸ ~+/-3.40 x 10 ³⁸	123.456、-3.4、-1.2E+12、 3.4E-3
LReal	64	+/-2.23 x 10 ⁻³⁰⁸ ~+/-1.79 x 10 ³⁰⁸	12345.123456789 -1.2e+40
Time	32	T#-24d_20h_31m_23s_648ms ~ T#24d_20h_31m_23s_647ms 格納值:2,147,483,648 ms ~ +2,147,483,647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	変数	0~254 バイトの大きさの文字 列	'ABC'

データタイプとしては使用できませんが、次の BCD 数字フォーマットも変換命令でサポートされています。

	サイズ(ビ ット)	数値の範囲	定数入力の例
BCD16	16	-999~999	123, -123
BCD32	32	-9999999~9999999	1234567, -1234567

実数のフォーマット

実数(浮動小数点数)は、ANSI/IEEE 754-1985 に規定されているように、32 ビット単精度数(Real)または 64 ビット倍精度数(LReal)として表されます。 単精度浮動小数点数の有効桁数は 6 桁、倍精度浮動小数点数の有効桁数は 15 桁です。 浮動小数点数を入力するときに、精度を保持するために有効桁数最大 6 桁(Real)または 15 桁(LReal)を指定することができます。

非常に大きな数値や小さな数値など、長い値が含まれている計算を行うと、計算結果が不正確になる可能性があります。 これは Real の場合 10 の x (x>6 (Real)、LReal の場合は 15 (LReal))乗倍異なる数値の場合に発生することがあります。 例(Real): 100 000 000 + 1 = 100 000 000.

3.3 データタイプ

文字データタイプのフォーマット

CPU は、1 バイト文字のシーケンスを格納するための STRING データタイプをサポートしています。 STRING データタイプには、合計文字カウント(文字列の文字数)と現在の文字カウントが格納されます。 STRING データタイプには、最大の合計文字カウント(1 バイト)、現在の文字カウント(1 バイト)、および最大 254 文字の各文字の格納に 1 文字あたり 1 バイトの最大 256 バイトが使用されます。

一重引用符('')を使用して、IN タイプの命令パラメータに対してリテラル文字列(定数)を使用することができます。 たとえば、'ABC'は、S_CONV 命令のパラメータ IN の入力として使用可能な 3 文字の文字列です。 また、OB、FC、FB、DB 用のブロックインターフェースエディタでデータタイプ[String]を選択することによって、文字列変数を指定することもできます PLC タグエディタで文字列を作成するとはできません。 文字列を宣言するときに、文字列のサイズをバイト単位で指定することができます。 たとえば、「MyString[10]」は MyString の最大サイズを 10 バイトに指定します。 最大サイズ指定子を括弧で囲んで指定しなかった場合は、254 を指定したものと見なされます。

次の例は、最大文字カウント 10、現在の文字カウント 3の STRING を定義します。つまり STRING には現在 1 バイト文字が 3 文字格納されていますが、1 バイト文字を 10 文字まで格納することができます。

合計文字カウント	現在の文字カウント	文字 1	文字2	文字3	 文字 10
10	3	'C' (16#43)	'A' (16#41)	'T' (16#54)	 -
バイト 0	バイト1	バイト2	バイト3	バイト4	 バイト 11

配列

基本のデータタイプの複数の要素を格納する配列を作成することができます。 配列は、OB、FC、FB、DB 用のブロックインターフェースエディタで作成することができます。 PLC タグエディタでは配列を作成するとはできません。

ブロックインターフェースエディタを使用して配列を作成するには、データタイプ [Array [lo .. hi] of type"]を選択して、[lo]、[hi]、[type]を次のように編集します。

- Io- 配列数の最小(最下位)の値
- hi- 配列数の最大(最上位)の値
- type-BOOL、SINT、UDINT など、基本のデータタイプの 1 つ

負の数もサポートされています。 ブロックインターフェースエディタの[Name]列に配列の名前を入力することができます。 ブロックインターフェースエディタでの配列の表示例を下の表に示します。

名前	データタイプ	コメント
My_Bits	Array [1 10] of BOOL	この配列には 10 のブール値が格納されています
My_Data	Array [-5 5] of SINT	この配列には 11 の SINT 値が格納されています(0 も含まれます)

プログラム内で配列の要素を参照するには次の構文を使用します。

• Array_name[*i*], - *i*で参照できます。

プログラムエディタでパラメータ入力として表示される例

- #My_Bits[3] 配列 My_Bits の 3 番目のビットを参照する
- #My_Data[-2] 配列 My_Data の 4 番目の SINT を参照する

#シンボルはプログラムエディタによって自動的に挿入されます。

3.3 データタイプ

DTL (Data and Time Long)日付と時刻データタイプ

DTL データタイプは、日付と時刻の情報が定義済みの 12 バイトの構造体です。 テンポラリメモリまたは DB のどちらかに DTL を定義することができます。

長さ(バイ ト)	フォーマット	値の範囲	値の入力例
12	時計とカレンダー (年-月タグ:時:分:秒.ナノ 秒)	最小: DTL#1970-01-01- 00:00:00.0 最大: DTL#2554-12-31- 23:59:59.999 999 999	DTL#2008-12-16- 20:30:20.250

DTL の各構成要素は、そのデータタイプと値の範囲が異なります。 指定する値のデータタイプは、対応する要素のデータタイプと一致している必要があります。

Byte	コンポーネン ト	データタイプ	値の範囲
0	年	UINT	1970~2554
1			
2	月	USINT	1~12
3	日	USINT	1~31
4	曜日	USINT	1(日曜日)~7(土曜日)
			週日は値の入力では考慮されません。
5	時	USINT	0~23
6	分	USINT	0~59
7	秒	USINT	0~59
8	ナノ秒	UDINT	0~999 999 999
9			
10			
11			

3.4 メモリカードを使用する

通知

CPUがサポートしているのはフォーマット済みの SIMATICメモリカード (ページ 443) のみです。 WindowsのフォーマットでSIMATICメモリカードをフォーマットしてしまうと、そのメモリカードは使用できなくなります。

フォーマット済みメモリカードにプログラムをコピーする前に、メモリカードに残っている以前のプログラムを消去する必要があります。

メモリカードはトランスファーカードまたはプログラムカードとして使用します。 メモリカードにコピーしたプログラムには、すべてのブロックコードとデータブロック、 テクノロジオブジェクト、デバイスコンフィグレーションが含まれています。 プログラムには強制値は含まれて**いません**。

- STEP 7 Basic を使用せずに CPU の内部ロードメモリにプログラムをコピーするにはトランスファーカードを使用します。 トランスファーカードを挿入すると、CPU の内部ロードメモリからユーザープログラムと強制値が最初に消去され、トランスファーカードから内部ロードメモリにプログラムがコピーされます。 転送処理が完了したら、トランスファーカードを抜き取る必要があります。
 - パスワードを失ったり忘れてしまった場合は (ページ 68)、空のトランスファーカードを使用して、パスワード保護されているCPUにアクセスします。 空のトランスファーカードを挿入すると、CPUの内部ロードメモリに存在するパスワード保護されたプログラムが消去され、 新しいプログラムをCPUにダウンロードすることができます。
- プログラムカードは、CPU の外部ロードメモリとして使用できます。プログラムカードを CPU に挿入すると、CPU の内部ロードメモリのすべて(ユーザープログラムと強制値)が消去され、外部ロードメモリ(プログラムカード)内のプログラムが実行されます。プログラムカードを装着した CPU にダウンロードを行うと、外部ロードメモリ(プログラムカード)のみが更新されます。

プログラムカードを挿入したときに CPU の内部ロードメモリが消去されていますから、プログラムカードは CPU に挿入したままにしておく必要があります。 プログラムカードを抜き取ると、CPU は STOP モードに移行します (エラーLED が点滅し、プログラムカードが抜き取られたことを示します)。

メモリカード上のプログラムには、プログラムブロック、データブロック、テクノロジオブジェクト、デバイスコンフィグレーションが含まれています。 メモリカードには強制値は含まれて**いません**。 強制値はプログラムの一部ではありませんが、CPU の内部ロードメモリであるか外部ロードメモリ(プログラムカード)であるかに関係なく、ロードメモリに格納されます。 プログラムカードを CPU に挿入すると、STEP 7 Basicは、プログラムカード上の外部ロードメモリにのみ強制値を適用します。

3.4 メモリカードを使用する

3.4.1 **CPU**にメモリカードを挿入する

警告

実行中の CPU にメモリカードを挿入すると(プログラムカードとして構成されているかトランスファーカードとして構成されているかに関係なく)、CPU は直ちに STOPモードに移行します。 制御装置が安全でない状態に陥り、被制御機器に予測できない動作が発生することがあります。 このような予測できない動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。 アプリケーションやプロセスには、緊急停止回路を必ず備えるようにしてください。

注意

静電放電によってメモリカードまたは CPU 側のスロットが損傷することがあります。

メモリカードを取り扱うときは、接地された導体パッドに触れるか、接地されたリストストラップを着用してください。 メモリカードは導電性のある容器に保管してください。

メモリカードを挿入するには、CPUのフロントカバーを開いて、所定のスロットにメモリカードを挿入します。 プッシュ - プッシュ形式のコネクタによって、簡単に挿抜できます。 メモリカードは、正しい方向でないと挿入できない構造になっています。





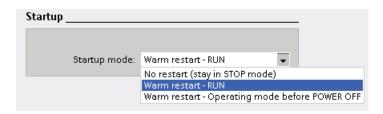
メモリカードが書き込み禁止になっていないことを確認します。 保護スイッチをスライドして「ロック」を解除します。

注記

CPU が STOP モードのときにメモリカードを挿入すると、診断バッファに、メモリカードの評価が開始されたことを示すメッセージが表示されます。 このメッセージは無視してください。 メモリカードの評価は、CPU が RUN モードに移行するまで開始されません。MRES を使用して CPU メモリをリセットするか、CPU の電源投入サイクルを実行します。

3.4.2 プロジェクトをメモリカードにコピーする前に**CPU**のスタートアップパラメータを設定する

トランスファーカードまたはプログラムカードにプログラムをコピーすると、そのプログラムには CPU のスタートアップパラメータも格納されます。 メモリカードにプログラムをコピーする前に、電源投入サイクルに続く CPU の動作モードが設定されていることを必ず確認します。 STOP モード、RUN モード、STARTUP モード(電源投入サイクルの前)のどのモードで CPU をスタートさせるかを選択します。



3.4.3 トランスファーカード

注意

静電放電によってメモリカードまたは CPU 側のスロットが損傷することがあります

メモリカードを取り扱うときは常に、接地された導体パッドに触れるか、接地された リストストラップを着用してください。 メモリカードは導電性のある容器に保管して ください。

トランスファーカードを作成する

トランスファーカードにプログラムをコピーする前に、CPUのスタートアップパラメータを設定する (ページ 83)のを忘れないように注意する必要があります。 トランスファーカードを作成するには次の手順を実行します。

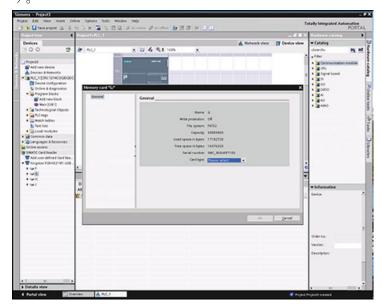
1. プログラミングデバイスに装着されているカードリーダ/ライタに空のメモリカード を挿入します。

(メモリカードが空でない場合は、Windows Explorer などのアプリケーションを使用して、メモリカード上の SIMATIC.S7S フォルダと S7_JOB.S7S ファイルを削除します)。

2. プロジェクトツリー(プロジェクトビュー)で、SIMATIC Card Reader フォルダを展開し、使用するカードリーダを選択します。

3.4 メモリカードを使用する

- 3. カードリーダ内のメモリカードを右クリックして、コンテキストメニューの [Properties]を選択し、[Memory card]ダイアログを表示します。
- 4. [Memory card]ダイアログで、ドロップダウンメニューから[Transfer]を選択します。 STEP 7 Basicによって空のトランスファーカードが作成されます。 CPUのパスワードを忘れてしまった (ページ 68)ために元に戻すときなど、空のトランスファーカードを作成する場合は、ここでカードリーダからトランスファーカードを抜き取ります。



- 5. プロジェクトツリーの CPU デバイス(PLC_1 [CPU 1214 DC/DC/DC])など)を選択し、メモリカードにドラッグすることによってプログラムを追加します。 (メモリカードに CPU デバイスをコピーアンドペーストする方法もあります)。 メモリカードに CPU デバイスをコピーすると、[Load preview]ダイアログが開きます。
- 6. この[Load preview]ダイアログで、[Load]ボタンをクリックすると、この CPU デバイスがメモリカードにコピーされます。
- 7. ダイアログに CPU デバイス(プログラム)が正常にロードされたことを示すメッセー ジが表示されたら、[Finish]ボタンをクリックします。

トランスファーカードを使用する

CPU にプログラムを転送するために次の手順を実行します。

- 1. CPUに (ページ 82)トランスファーカードを挿入します。 CPUがRUNモードであった場合はSTOPモードに移行します。 (メンテナンスLEDが点滅し、メモリカードの評価が必要であることを示します)。
- 2. メモリカードの評価を行うために次のいずれかを実行します。
 - CPU の電源を切った後に再投入する
 - RUN モードから STOP モードへの移行
 - メモリリセット(MRES)を実行します。
- 3. リブートしてメモリカードの評価を行った後で、CPU は内部ロードメモリにプログラムをコピーします。 コピー操作が完了すると、メンテナンス LED が点滅し、トランスファーカードを抜き取ることができることを示します。
- 4. CPU からトランスファーカードを抜き取ります。
- 5. 内部ロードメモリに転送された新しいプログラムの評価を行うために、次のいずれ かを実行します。
 - CPU の電源を切った後に再投入する
 - RUN モードから STOP モードへの移行
 - メモリリセット(MRES)を実行します。

CPU は、このプロジェクト用に設定されているスタートアップモード(**RUN** モードまたは **STOP** モード)に移行します。

注記

CPU を RUN モードに設定する前に、トランスファーカードを抜き取る必要があります。

3.4 メモリカードを使用する

3.4.4 プログラムカード

注意

静電放電によってメモリカードまたは CPU 側のスロットが損傷することがあります。

メモリカードを取り扱うときは、接地された導体パッドに触れるか、接地されたリストストラップを着用してください。 メモリカードは導電性のある容器に保管してください。



メモリカードが書き込み禁止になっていないことを確認します。 保護スイッチをスライドして「ロック」を解除します。

プログラムカードにプログラム要素をコピーする前に、メモリカード に残っている以前のプログラムを消去する必要があります。

プログラムカードを作成する

メモリカードは、プログラムカードとして使用すると CPU の外部ロードメモリになります。 プログラムカードを抜き取ると、CPU の内部ロードメモリは空になっています。

注記

CPU に空のメモリカードを挿入し、CPU の電源投入サイクル、STOP モードから RUN モードへの移行、メモリのリセット(MRES)のいずれかを実行することによってメモリカードの評価を行った場合、CPU の内部ロードメモリ内のプログラムおよび強制値がメモリカードにコピーされます。 (メモリカードがプログラムカードになります)。 コピーの完了後に、CPU の内部ロードメモリ内のプログラムが消去されます。 次に、CPU は設定されているスタートアップモード(RUN モードまたは STOP モード)に移行します。

プログラムカードにプロジェクトをコピーする前に、CPUのスタートアップパラメータを設定する (ページ 83)のを忘れないように注意する必要があります。 STEP 7 Basicを使用してプログラムカードを作成するには次の手順を実行します。

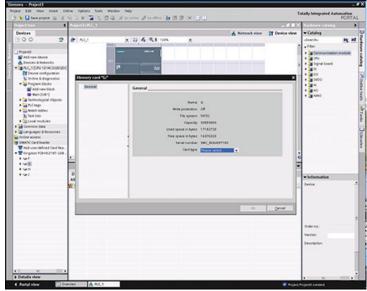
1. プログラミングデバイスに装着されているカードリーダ/ライタに空のメモリカード を挿入します。

(メモリカードが空でない場合は、Windows Explorer などのアプリケーションを使用して、メモリカード上の SIMATIC.S7S フォルダと S7_JOB.S7S ファイルを削除します)。

- 2. プロジェクトツリー(プロジェクトビュー)で、SIMATIC Card Reader フォルダを展開し、使用するカードリーダを選択します。
- 3. カードリーダ内のメモリカードを右クリックして、コンテキストメニューの [Properties]を選択し、[Memory card]ダイアログを表示します。

3.4 メモリカードを使用する





- 5. プロジェクトツリーの CPU デバイス(PLC_1 [CPU 1214 DC/DC/DC])など)を選択し、メモリカードにドラッグすることによってプログラムを追加します。 (メモリカード に CPU デバイスをコピーアンドペーストする方法もあります)。 メモリカードに CPU デバイスをコピーすると、[Load preview]ダイアログが開きます。
- 6. この[Load preview]ダイアログで、[Load]ボタンをクリックすると、この CPU デバイスがメモリカードにコピーされます。
- 7. ダイアログに CPU デバイス(プログラム)が正常にロードされたことを示すメッセージが表示されたら、[Finish]ボタンをクリックします。

プログラムカードを CPU のロードメモリとして使用する

注意

CPU に空のメモリカードを挿入すると、CPU は STOP モードに移行します。 CPU の電源投入サイクル、STOP モードから RUN モードへの移行、またはメモリのリセット(MRES)を実行すると、CPU の内部ロードメモリがメモリカードにコピーされ(メモリカードがプログラムカードとして設定される)、内部ロードメモリからプログラムが消去されます。 プログラムカードを抜き取ると、CPU の内部ロードメモリにはプログラムが存在しない状態になります。

CPU でプログラムカードを使用するには次の手順を実行します。

- 1. CPU にプログラムカードを挿入します。 CPU が RUN モードであった場合は STOP モードに移行します。 メンテナンス LED が点滅し、プログラムカードの評価が必要 であることを示します。
- 2. プログラムカードの評価を行うために次のいずれかを実行します。
 - CPU の電源を切った後に再投入する
 - RUN モードから STOP モードへの移行
 - メモリリセット(MRES)を実行します。
- 3. CPU は自動的にリブートします。 リブートしてプログラムカードの評価を行った後で、CPU は内部ロードメモリを消去します。

CPU は、この CPU 用に設定されているスタートアップモード(RUN モードまたは STOP モード)に移行します。

プログラムカードを CPU に挿入したままにしておく必要があります。 プログラムカードを抜き取ると、CPU の内部ロードメモリにはプログラムが存在しない状態になります。

个警告

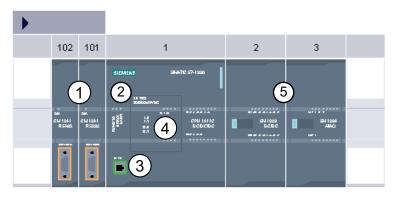
プログラムカードを抜き取った場合、内部ロードメモリの内容が失われ、CPU はエラーを生成します。 CPU は STOP モードに移行し、エラーLED が点滅します。

制御装置が安全でない状態に陥り、被制御機器に予測できない動作が発生することがあります。 このような予測できない動作によって、死傷事故などの重大な人的傷害や物的損害が発生する可能性があります。

3.4 メモリカードを使用する

デバイスコンフィグレーション

プロジェクトに CPU および付加モジュールを追加して、PLC 用のデバイスコンフィグレーションを作成します。



- ① 通信モジュール(CM): 最大 3 モジュール、スロット 101、102、103 に挿入
- ② CPU: スロット1
- ③ CPU のイーサネットポート
- ④ シグナルボード(SB): 最大 1 枚、CPU に挿入
- ⑤ デジタルまたはアナログ I/O 用シグナルモジュール(SM): 最大 8 モジュール、スロット 2 ~ 9 に 挿入 (CDL 4244C の場合は 8 モジュール CDL 4244C の場合は 8 モジュール CDL 4244C の場合

(CPU 1214C の場合は 8 モジュール、CPU 1212C の場合は 2 モジュール、CPU 1211C の場合は使用不可)

デバイスコンフィグレーションを 作成するにはプロジェクトにデバ イスを追加します。

- ポータルビューで、[Devices Networks]を選択して[Add device]をクリックします。
- プロジェクトビューで、プロジェクト名の下の[Add device] をダブルクリックします。



4.1 CPU を挿入する

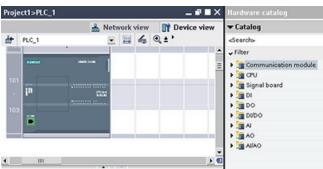
4.1 CPU を挿入する

プロジェクトに CPU を挿入することによりデバイスコンフィグレーションを作成します。 [Add a new device]ダイアログで CPU を選択し、ラックおよび CPU を作成します。

[Add a new device]ダイアロ



ハードウェアコンフィグレー ションのデバイスビュー



デバイスビューで CPU を選択すると、インスペクタウィンドウにその CPU のプロパティが表示されます。



注記

CPUのIPアドレスは未設定です。デバイスコンフィグレーションで、CPUにIPアドレスを手動で割り付ける必要があります。 CPU がネットワーク上のルータに接続されている場合は、そのルータ用のIPアドレスも入力する必要があります。

4.2 未指定の CPU 用構成を検出する

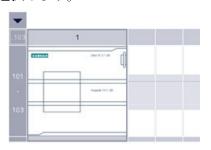
既存のハードウェア構成をアップロードするのは簡単です



CPU に接続している場合は、すべてのモジュールを含め、その CPU の構成をプロジェクトにアップロードすることができます。 新しいプロジェクトを作成し、特定の CPU ではなく [unspecified CPU]を選択します。 ([First steps]から[Create a PLC program]を選択することによって、デバイスコンフィグレーションの全体をスキップすることもできます。 この場合、STEP 7 Basic によって未指定の CPU が自動的に作成されます)。

プログラムエディタで、[Online]メニューから[Hardware detection]を選択します。

デバイスコンフィグレーションエディタで、接続デバイスの構成検出用のオプション を選択します。





オンラインダイアログで CPU を選択すると、STEP 7 Basic は、その CPU から、すべてのモジュール(SM、SB、CM)を含めてハードウェア構成をアップロードします。 CPU とモジュールのパラメータを構成することができます。





4.3 CPU の動作を構成する

4.3 CPU の動作を構成する

CPU の動作パラメータを設定するには、デバイスビューで CPU を選択し(CPU の周囲 に青色の枠が表示されます)、インスペクタウィンドウの[Properties]タブを使用します。



プロパティを編集して、下記のパラメータを設定します。

- PROFINET interface (PROFINET インターフェース): CPU の IP アドレスと時刻の同期 を設定します。
- DI, DO, and AI (DI、DO、および AI): ローカル(オンボード)のデジタルおよびアナログI/O の動作を構成します。
- High-speed counters (高速カウンタ)および Pulse generators (パルスジェネレータ): 高速カウンタ(HSC)とパルス列操作(PTO)およびパルス幅変調(PWM)に使用するパルスジェネレータを有効にして構成します。

CPU またはシグナルボードの出力をパルスジェネレータとして設定する(PWM または基本モーションコントロール命令と共に使用する)場合、対応する出力アドレス(Q0.0、

Q0.1、Q4.0、Q4.1)はQメモリから削除され、ユーザープログラムの他の目的に使用することはできません。 ユーザープログラムがパルスジェネレータとして使用されている出力に値を書き込むと、その CPU はこの値を物理出力に書き込みません。

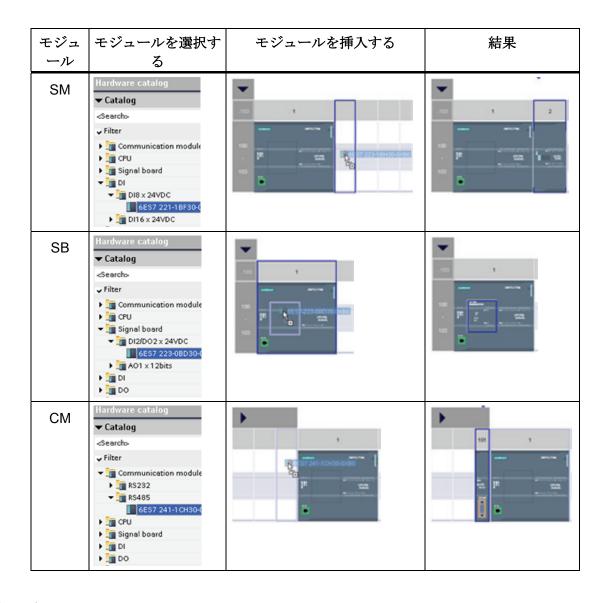
- Startup (スタートアップ): STOP モードでのスタートやウォームリスタート後の RUN モードへの移行など、オフからオンへの移行後の CPU の動作を選択します。
- Time of day (時刻): 時刻、タイムゾーン、サマータイムを設定します。
- Protection (保護): この CPU の読み取り/書き込み保護およびアクセス用パスワードを設定します。
- System and clock memory (システムメモリとクロックメモリ): 「システムメモリ」機能用のバイト(「first-scan (最初のスキャン)」ビット、「always-on (常にオン)」ビット、「always-off (常にオフ)」ビット)を有効設定にし、「クロックメモリ」ファンクション用のバイト(各ビットが定義されている周期で交互にオン/オフを繰り返す)を有効にします。
- Cycle time (サイクルタイム): 最大サイクルタイムまたは最小固定スキャンサイクルタイムを定義します。
- Communications load (通信負荷): 通信タスクに割り付ける CPU 時間の割り合いを指定 します。

4.4 構成にモジュールを追加する

CPU にモジュールを追加するには、ハードウェアカタログを使用します。 モジュール には次の3種類があります。

- シグナルモジュール(SM) デジタルまたはアナログ I/O を追加するために使用します。 シグナルモジュールは CPU の右側に接続します。
- シグナルボード(SB) CPU に少数の I/O ポイントを追加するために使用します。 SB は CPU の前面に装着します。
- 通信モジュール(CM) CPU に通信ポート(RS232 または RS485)を追加するために 使用します。 通信モジュールは CPU の左側に接続します。

ハードウェア構成にモジュールを挿入するには、ハードウェアカタログでモジュールを 選択し、そのモジュールをダブルクリックするか、強調表示されているスロットにドラ ッグします。

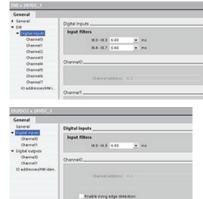


4.5 モジュールのパラメータを構成する

モジュールの動作パラメータを設定するには、デバイスビューでモジュールを選択し、インスペクタウィンドウの[Properties]タブを使用して、モジュールのパラメータを設定します。

シグナルモジュール(SM)またはシグナルボード(SB)を設定する

- Digital I/O (デジタル I/O): 入力は、立ち上がりエッジ検出または立ち下がりエッジ検出の設定をすることができます(それぞれをイベントおよびハードウェア割り込みに関連付けて)。また、次の入力プロセスイメージの更新まで「パルスキャッチする」(パルス検出後オンの状態を保持する)ように設定することもできます。 出力には値の維持または代替値の設定をすることができます。
- Analog I/O (アナログ I/O): それぞれの入力に対して、計測のタイプ(電圧または電流)、範囲および平滑化、アンダーフローまたはオーバーフローを有効にするなどのパラメータを設定します。 出力には、出力タイプ(電圧または電流)や、診断のための短絡(電圧出力用)や診断の基準の上限/下限などのパラメータが用意されています。

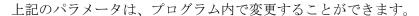


• I/O diagnostic addresses (I/O 診断アドレス): モジュールの入力および出力のセット に対する開始アドレスを設定します。

4.5 モジュールのパラメータを構成する

通信モジュール(CM)を設定する

- Port configuration (ポートの設定): ボーレート、 パリティ、データビット、ストップビット、フロー制御、XON および XOFF 文字、待ち時間などの通信パラメータを設定します。
- Transmit message configuration (送信メッセージのコンフィグレーション): 通信関連のオプションを有効にして設定します。
- Receive message configuration (受信メッセージ のコンフィグレーション): message-start (メッ セージ開始)および message-end (メッセージ終 了)パラメータを有効にして設定します。

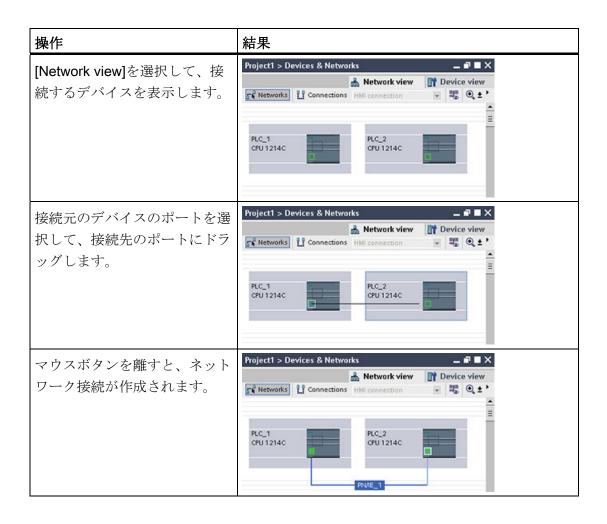




4.6 ネットワーク接続を作成する

4.6 ネットワーク接続を作成する

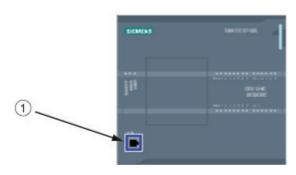
プロジェクト内のデバイス間のネットワーク接続を作成するには、デバイスコンフィグレーションの[Network view]を使用します。 ネットワーク接続を作成し、インスペクタウィンドウの[Properties]タブを使用して、ネットワークのパラメータを構成します。



4.7 IP アドレスをプロジェクトで設定する

PROFINET インターフェースを設定する

CPUをラック上に構成 (ページ 94)したら、PROFINETインターフェース用のパラメータを設定することができます。 CPU上の緑色の[PROFINET]ボックスをクリックし、 [PROFINET port]を選択します。 インスペクタウィンドウの[Properties]タブに、 PROFINETポートが表示されます。



① PROFINET ポート

IPアドレスを設定する

イーサネット(MAC)アドレス: PROFINET ネットワークの各デバイスには、製造メーカーによって識別用のメディアアクセス制御アドレス(MAC アドレス)が割り付けられています。 MAC アドレスは、2 桁の 16 進数を伝送順にハイフン(-)またはコロン(:)で区切った 6 つのグループで構成されています(01-23-45-67-89-AB や 01:23:45:67:89:AB など)。

IP address: また、各デバイスにはインターネットプロトコル(IP)アドレスも必要です。 このアドレスを使用して、デバイスはさらに複雑にルーティングされたネットワーク上 にデータを配信することができます。

IP アドレスは 8 ビットのセグメントに分割され、ドット付き 10 進形式で表されます (211.154.184.16 など)。 IP アドレスの最初の部分はネットワーク ID (ネットワークの 識別)で、2 番目の部分はホスト ID (ネットワーク上のそれぞれのデバイスに固有)です。 IP アドレス 192.168.x.y は、インターネット上でルーティングされていないプライベートネットワークの一部として認識されるアドレスの標準指定です。

4.7 IP アドレスをプロジェクトで設定する

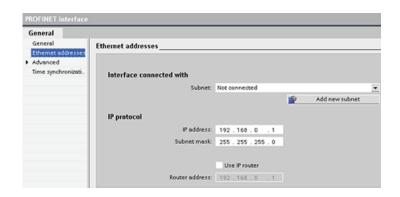
Subnet mask: サブネットは、接続されているネットワークデバイスの論理的な分類です。 サブネット上のノードは、通常は、ローカルエリアネットワーク(LAN)上の相互に物理的に近い場所に配置されます。 マスク(サブネットマスクまたはネットワークマスクとも呼ばれる)は、IP サブネットの境界を定義します。

サブネットマスク 255.255.255.0 は、一般的に小規模ローカルネットワークに適しています。 つまり、このネットワーク上のすべての IP アドレスは最初の 3 オクテットが同じで、このネットワーク上のさまざまなデバイスは最後のオクテット(8 ビットのフィールド)で識別されます。 たとえば、サブネットマスク 255.255.255.0 と IP アドレス 192.168.2.0~192.168.2.255 を小規模ローカルネットワーク上のデバイスに割り付けます。

異なるサブネット間の接続のみがルータを経由します。 サブネットを使用する場合は、IP ルータが必要です。

IP router: ルータは LAN 間を接続します。 LAN 内のコンピュータは、ルータを使用して、他のネットワークにメッセージを送信することができます。そのネットワークの背後に、また他の LAN が存在する可能性もあります。 データの送信先が LAN 内に存在しない場合、ルータは、そのデータを送信先に配信することができる他のネットワークまたはネットワークのグループに転送します。

ルータはIPアドレスを使用して、データパケットの送受信を行います。



IP addresses プロパティ: [Properties]ウィンドウで[Ethernet address]設定エントリを選択します。 TIA ポータルに [Ethernet address configuration]ダイアログが表示されます。このダイアログで、ソフトウェアプロジェクトを受信する CPU の IP アドレスの関連付けを行います。

注記

CPUのIPアドレスは未設定です。IPアドレスを手動でCPUに割り付ける必要があります。CPUがネットワーク上のルータに接続されている場合は、そのルータのIPアドレスも入力する必要があります。プロジェクトをダウンロードすると、すべてのIPアドレスが設定されます。

詳細については、「プログラミングデバイスおよびネットワークデバイスにIPアドレスを割り付ける (ページ 298)」を参照してください。

IPアドレス用パラメータの定義を下の表に示します。

パラメータ		説明
Subnet	new subnet]ボタン 次の 2 種類の接続。 • デフォルトの[N	ナブネット名。 新しいサブネットを作成するには[Addをクリックする。 [Not connected]がデフォルト。 タイプが指定できる。 ot connected]によるローカル接続 に2つ以上のデバイスが存在する場合は、サブネット
	が必要	
IP protocol	IP address	CPU に割り付ける IP アドレス
	Subnet mask	割り付けるサブネットマスク
	Use IP router	IP ルータを使用するには、このチェックボックスを オンにする
	Router address	ルータを使用する場合にルータに割り付ける IP アドレス

デバイスコンフィグレーション 4.7 IP アドレスをプロジェクトで設定する

5.1 PLC システム設計用ガイドライン

PLC システムを設計するとき、さまざまな方法および基準を選択することができます。 多くの設計プロジェクトに適用可能な一般的なガイドラインを下記に示します。 社内の手続きや基準および独自の研修および立場において認められている慣例に従うのは当然です。

推奨手順	タスク
プロセスまたは機器 の細分化	相互に独立しているレベルにプロセスまたは機器を細分化します。 この細分化はコントローラ間の境界を決定し、機能の記述仕様およびリソースの割り付けに影響を与えます。
機能仕様の作成	I/O ポイント、操作の機能記述、各アクチュエータ(ソレノイド、モータ、ドライブなど)の操作を許可する前に達成しなければならない状態など、プロセスまたは機器のそれぞれのセクションの動作の記述、オペレータインターフェースおよびプロセスおよび機器の他のセクションとのすべてのインターフェースの記述を作成します。
安全回路の設計	安全のためのハードワイヤードロジックが必要な機器をリストアップします。 制御デバイスは安全でない状態に陥る可能性があり、機械装置の予測できない起動や動作の変化が発生することがあります。 機械装置の予測できない動作または誤動作によって、重大な人的傷害や物的損害が発生する可能性があるため、このような事態の発生を防止するための電気機械的補助装置(PLC とは独立して動作する)の実装を考慮します。安全回路の設計に下記のタスクを含める必要があります。

5.1 PLC システム設計用ガイドライン

推奨手順	タスク
オペレータステーシ	機能仕様の要件に基づいて、下記のオペレータステーションの図を作成する。
ョンを指定する	それぞれのオペレータステーションの場所とプロセスまたは機器との関連を示
	す概要図。
	• ディスプレイ、スイッチ、照明など、オペレータステーション用デバイスの機
	器の配置図。
	• PLC およびシグナルモジュールの I/O と関連付けられている配線図。
構成図を作成する	機能仕様の要件に基づいて、制御設備の構成図を作成する。
	• それぞれの PLC の場所とプロセスまたは機器との関連を示す概要図。
	• それぞれの PLC および I/O モジュールの機器の、キャビネットや他の設備を含
	めた配置図。
	• それぞれの PLC および I/O モジュールの、デバイスの型式番号、通信アドレ
	ス、および I/O アドレスを含めた配線図。
シンボリック名の一	絶対アドレスに対するシンボリック名の一覧表を作成する。 物理的な I/O 信号だ
覧表を作成する	けでなく、プログラム内で使用する他の要素(タグ名など)も含める。

5.2 プログラムを構造化する

オートメーションタスク用のユーザープログラムを作成するには、そのプログラム用の 命令をプログラムブロックに挿入します。

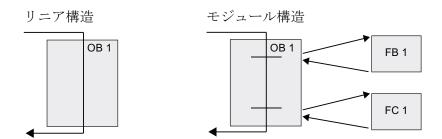
- オーガニゼーションブロック(OB)は CPU 内の特定のイベントに応答し、ユーザープログラムの実行に割り込むことができます。 ユーザープログラムのサイクリック 実行用のデフォルトのオーガニゼーションブロック(OB 1)は、ユーザープログラム 用の基本構造を備えた、ユーザープログラムに必要な唯一のプログラムブロックです。 プログラム内に OB を含めておくと、その OB によって OB 1 の実行に割り込みが発生します。他の OB は、スタートアップタスクなど、割り込みおよびエラーを処理するか、所定の間隔で所定のプログラムコードを実行するために所定の機能を実行します。
- ファンクションブロック(FB)は、他のプログラムブロック(OB、FB、FC)から呼び出されて実行されるサブルーチンです。 呼び出し元のブロックは FB にパラメータを渡し、所定の呼び出しまたはその FB のインスタンス用のデータを格納するデータブロック(DB)を指定します。 インスタンス DB を変更することによって、汎用のFB で一連のデバイスの動作を制御することができます。 たとえば、1 つの FB で、それぞれのポンプまたはバルブ固有の動作パラメータを格納した異なるインスタンス DB を使用して、複数のポンプまたはバルブを個別に制御することができます。
- ファンクション(FC)は、他のプログラムブロック(OB、FB、FC)から呼び出されて 実行されるサブルーチンです。 FC には、関連付けられているインスタンス DB は 存在しません。 呼び出し元のブロックは FC にパラメータを渡します。 FC からの 出力値をメモリアドレスまたはグローバル DB に書き込む必要があります。

5.2 プログラムを構造化する

ユーザープログラムの構造を選択する

アプリケーションの要件に基づいてリニア構造またはモジュール構造のどちらかを選択し、 ユーザープログラムを作成することができます。

- リニアプログラムは、オートメーションタスクのすべての命令を 1 つずつ順番に実行します。 通常は、リニアプログラムでは、すべてのプログラム命令をプログラムのサイクリック実行用 OB (OB 1)に配置します。
- モジュールプログラムは、特定のタスクを実行するプログラムブロックを呼び出します。 モジュール構造を作成するには、プロセスの技術的機能に対応した小さなタスクに、複 雑なオートメーションタスクを細分化します。 それぞれのプログラムブロックが、それ ぞれの細分化されたタスクに対するプログラムセグメントを構成します。 プログラムブ ロックから他のプログラムブロックを呼び出すことによって、プログラムを構造化しま す。



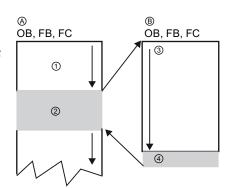
ユーザープログラム内で再利用のできる汎用プログラムブロックを作成することによって、 ユーザープログラムの設計および実装を単純化することができます。 汎用プログラムブロックを使用することによって、さまざまなメリットが得られます。

- ポンプやモータの制御など、再利用のできる標準タスク用のプログラムブロックを作成することができます。 また、このような汎用プログラムブロックをライブラリに格納しておき、さまざまなアプリケーションやソリューションで使用することもできます。
- 機能タスクに関連付けられたモジュール式コンポーネントを使用してユーザープログラムを構造化すると、プログラムデザインの理解および管理が簡単になります。 モジュール式コンポーネントは、プログラム設計を標準化するのに役立つだけではなく、プログラムコードの更新または修正を短時間で簡単に実行できるようするためにも役立ちます。
- モジュール式コンポーネントを作成することによって、プログラムのデバッグが単純化されます。 完全なプログラムを一連のモジュール式プログラムセグメントとして構造化することによって、それぞれのプログラムブロックを開発した時点で機能試験を実行することができます。
- 特定の技術機能に関連付けられたモジュール式コンポーネントを作成することによって、 完成したアプリケーションの試運転が単純化され所要時間の短縮に役立ちます。

5.3 ブロックを使用してプログラムを構造化する

FB および FC を汎用のタスクを実行するように設計することによって、モジュール式のプログラムブロックを作成します。 このように再利用できるモジュールを他のプログラムブロックから呼び出すようにして、プログラムを構造化します。 呼び出し元のブロックから呼び出したブロックにデバイス固有のパラメータを渡します。

- A 呼び出し元のブロック
- B 呼び出された(または割り込みを行う)ブロック
- ① プログラムの実行
- ② 他のブロックを呼び出す操作
- ③ プログラムの実行
- ④ ブロックの終り(呼び出し元のブロックに 戻る)

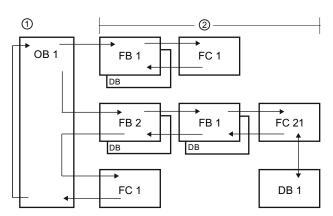


プログラムブロックから他のプログラムブロックを呼び出すと、CPU は呼び出されたブロック内のプログラムコードを実行します。 呼び出されたブロックの実行が終了すると、CPU は呼び出し元のブロックの実行に復帰します。

処理は、ブロック呼び出しの後 に続いている命令の実行に移り ます。

ブロック呼び出しをネストして モジュール構造を階層化するこ とができます。

- ① サイクルの開始
- ② ネストレベル



5.3 ブロックを使用してプログラムを構造化する

再利用できるプログラムブロックを作成する



OB、FB、FC、グローバル DB を作成するには、プロジェクト ナビゲータの[Program blocks]に ある[Add new block]ダイアログ を使用します。

プログラムブロックを作成するときは、そのブロック用のプログラミング言語を選択します。 DBの場合はデータを格納するだけですから、プログラミング言語の選択は不要です。

5.3.1 オーガニゼーションブロック(OB)

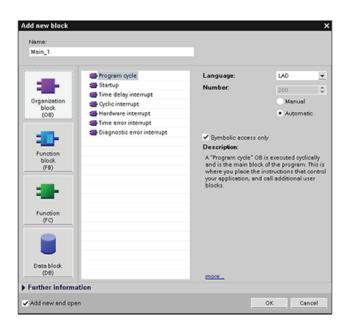
オーガニゼーションブロックは、プログラム用の構造を備えています。 オペレーティングシステムとユーザープログラム間のインターフェースとして機能します。 OB はイベント駆動です。 診断割り込みや周期割り込みなどのイベントが発生すると、CPU は所定の OB を実行します。スタートイベントおよび動作が定義済みになっている OB も存在します。

作成したメインプログラムをプログラムサイクル OB に格納します。 ユーザープログラム内に複数のプログラムサイクル OB を含めることができます。 RUN モードでは、プログラムサイクル OB は最低のレベルの優先度で実行され、他のすべてのタイプのプログラム処理が割り込むことができます。 スタートアップ OB は RUN モードに移行する前に実行されるため、スタートアップ OB がプログラムサイクル OB に割り込むことはありません。

CPU は、プログラムサイクル OB の処理を完了すると、そのプログラムサイクル OB の実行を直ちに繰り返します。 この反復処理は、プログラマブルロジックコントローラに対して使用される「通常」の処理です。 多くのアプリケーションでは、ユーザープログラム全体が単一のプログラムサイクル OB 内に配置されます。

割り込みおよびエラーの処理や所定の周期で特定のプログラムコードを実行するために、スタートアップタスクなどの特定の機能を実行する OB を作成することができます。これらの OB はプログラムサイクル OB の実行に割り込みを発生します。

ユーザープログラム内に新しい OB を作成するには[Add new block]ダイアログを使用します。



それぞれの優先度レベルに基づいて OB は他の OB に割り込むことができます。割り込みの処理は常にイベント駆動によって行われます。このようなイベントが発生すると、CPU はユーザープログラムの実行を中断して、そのイベントを処理するように構成されている OB を呼び出します。割り込んだ OB の実行が終了すると、CPU は、ユーザープログラムの実行を割り込み発生時点から再開します。

CPUは、それぞれのOBに割り付けられている優先度に基づいて割り込みイベントの処理順序を決定します。あらかじめ各イベントには所定の優先度が割り付けられています。いくつかの割り込みイベントを優先度クラスにまとめることができます。 詳細については、ユーザープログラムの実行の「PLCのコンセプト」 (ページ 43)を参照してください。

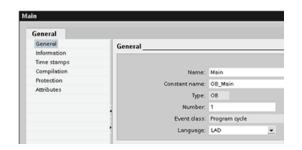
5.3 ブロックを使用してプログラムを構造化する

OB のクラス内に OB を追加作成する

ユーザープログラム用、さらにはプログラムサイクルクラス、およびスタートアップ OB クラス用の OB を複数作成することができます。 OB を作成するには[Add new block]ダイアログを使用します。作成する OB の名前と 200 番以上の OB 番号を入力します。

ユーザープログラム用に複数のプログラムサイクル OB を作成すると、CPU はそれぞれのプログラムサイクル OB をメインプログラムサイクル OB (デフォルトでは OB 1)から始めて、番号順に実行します。 たとえば、CPU は、最初のプログラムサイクル OB (OB 1)が終了すると、次のプログラムサイクル OB (OB 200 など)を実行します。

OB の動作を設定する



OB の動作パラメータを修正することができます。たとえば、遅延 OB または周期 OB の時間パラメータを設定することができます。

5.3.2 ファンクション(FC)

ファンクション(FC)は、一般に、一連の入力値に基づいて所定の動作を実行するプログラムブロックです。 FC は、その動作の結果をメモリロケーションに格納します。

下記のタスクを実行するには FC を使用します。

- 数学的計算など、標準的で再使用可能な操作を実行する
- ビット論理演算を使用して、個別制御などの技術的機能を実行する

FC は、プログラム内のさまざまなポイントで何回も呼び出すことができます。 FC を 反復使用することによって、頻繁に反復されるタスクのプログラミングが単純化されます。

FCには、関連付けられているインスタンスデータブロック(DB)は存在しません。 計算に使用するテンポラリデータにはローカルデータスタックが使用されます。 このテンポラリデータは保存されません。 このデータを保存するには、出力値を M メモリなどのグローバルメモリロケーションまたはグローバル DB に格納します。

5.3.3 ファンクションブロック(FB)

ファンクションブロック(FB)は、パラメータおよび静的データ用としてインスタンスデータブロックを使用するプログラムブロックです。 FB には、データブロック(DB)に格納されている可変のメモリ、つまり「インスタンス」DB が存在します。 インスタンスDB には、FB のインスタンス(つまり呼び出し)に関連付けられたメモリブロックが用意され、FB の終了後にデータが格納されます。 FB の呼び出しごとに異なるインスタンスDB を割り付けることができます。 インスタンスDB を使用して、1 つの汎用 FB を複数のデバイスの制御に使用することができます。 1 つのプログラムブロックで1 つのFB と 1 つの DB を呼び出すことによってプログラムを構造化します。 このようにすれば、CPU は、その FB 内のプログラムコードを実行して、インスタンス DB にブロックパラメータと静的ローカルデータを格納します。 FB の実行が終了すると、CPU は、その FB を呼び出した元のプログラムブロックの実行に戻ります。 インスタンス DB には、その FB のインスタンスの値が保持されます。 これらのインスタンス値は、同一のスキャンサイクルまたは他のスキャンサイクルで、この FB の呼び出し以降で使用することができます。

再利用できる割り付けメモリ付きプログラムブロック

通常は、1スキャンサイクルで動作が完了しないタスクまたはデバイスの動作を制御するために1つのFBを使用します。動作パラメータを保存しておいて次のスキャンから迅速にアクセスできるようにするために、ユーザープログラム内の各FBは複数のインスタンスDBを使用します。FBを呼び出すときに、FBのその呼び出し、つまり「インスタンス」のためのブロックパラメータと静的ローカルデータを格納するインスタンスDBを指定します。インスタンスDBには、FBが実行を終了した後の、これらの値が保持されます。

汎用制御タスク用の FB を設計することによって、FB の呼出しごとに異なるインスタンス DB を選択し、その FB を複数のデバイスに対して再利用することができます。

FB は、インスタンス DB に、入力(IN)、出力(OUT)、入出力(IN_OUT)の各パラメータを保存します。

5.3 ブロックを使用してプログラムを構造化する

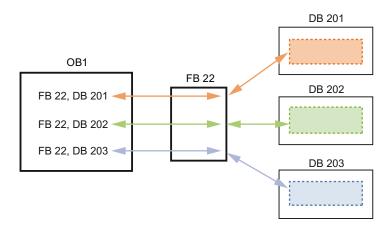
初期値を代入する

ファンクションブロック(FB)の入力、出力、入出力のパラメータに値が代入されていない場合は、インスタンスデータブロック(DB)に格納されている値が使用されます。 ただし、パラメータに値を代入しなければならない場合もあります。

FB インターフェースで、パラメータに初期値を代入することができます。 これらの値 は関連付けられている DB に転送されます。 パラメータに初期値を代入しなかった場合は、インスタンス DB に格納されている現在値が使用されます。

単一のFBと複数のDBを使用する

呼出しごとに異なるデータブロックを使用して、同一の FB を 3 回呼び出す OB の例を下の図に示します。 この構造を使用して、それぞれのデバイスのための呼び出しごとに異なるインスタンスデータブロックを割り付けることによって、複数のモータなど、類似した複数のデバイスを 1 つの汎用 FB で制御することができます。 それぞれの DB には、個別のデバイスのデータ(速度、起動時間、総運転時間など)が格納されます。 この例では、FB 22 が 3 つのデバイスを制御し、DB 201 には最初のデバイスの運転データが格納され、DB 202 には 2 番目のデバイスの運転データが格納され、DB 203 には 3 番目のデバイスの運転データが格納されます。



5.3.4 データブロック (DB)

プログラムブロック用のデータを保存するために、ユーザープログラム内にデータブロック(DB)を作成することができます。 ユーザープログラム内のすべてのプログラムブロックはグローバル DB 内のデータにアクセスできますが、インスタンス DB には特定のファンクションブロック(FB)のデータが格納されます。 DB を読み取り専用として定義することができます。

DB に格納されているデータは、関連付けられているプログラムブロックが終了しても削除されません。 DB には次の 2 種類があります。

- グローバル DB プログラム内のプログラムブロック用のデータが格納されます。 グローバル DB 内のデータには、すべての OB、FB、FC からアクセスすることができます。
- インスタンス DB 特定の FB 用のデータが格納されます。 インスタンス DB 内のデータ構造は、その FB のパラメータ(Input、Output、InOut)と静的データを反映しています。 (FB 用のテンポラリメモリはインスタンス DB には保存されません)。

注記

インスタンス DB は関連の FB 用のデータを反映していますが、どのプログラムブロックからもインスタンス DB 内のデータにアクセスすることができます。

5.4 データの一貫性を理解する

CPU はすべての基本的なデータタイプおよびすべてのシステム定義構造体(IEC_TIMERS や DTL など)のデータの一貫性を保持します。 値の読み取りまたは書き込みに割り込むことはできません。 (たとえば、4 バイトの DWord の読み取りまたは書き込みが完了するまでは、その DWord 値へのアクセスは禁止されます)。 プログラムサイクル OB および割り込み OB が同時に同一のメモリロケーションに書き込むことができないようにするために、CPU は、プログラムサイクル OB 内の読み取りまたは書き込み操作が完了するまでは、割り込み OB を実行しません。

ユーザープログラム内でプログラムサイクル OB と割り込み OB 間でメモリ内の値を共有する場合は、ユーザープログラムで、これらの値の修正または読み取りが正しく行われるように保証する必要があります。 プログラムサイクル OB 内で DIS_AIRT および EN_AIRT 命令を使用して、共有値へのアクセスを保護することができます。

- 読み取りまたは書き込み操作を実行しているときに割り込み OB が実行されないように するために、プログラムブロックに DIS_AIRT 命令を挿入します。
- 割り込み OB によって変更される可能性のある値の読み取りまたは書き込みを行う命令 を挿入します。
- DIS_AIRT をキャンセルして割り込み OB を実行できるように、シーケンスの終りに EN AIRT 命令を挿入します。

HMI デバイスまたは他の CPU からの通信要求もプログラムサイクル OB の実行に割り込むことができます。したがって、通信要求もデータの一貫性の問題の原因になる可能性があります。 CPU は、ユーザープログラム命令による基本的なデータタイプの読み書きが常に一貫性を保持して行われるようにしています。 ユーザープログラムは通信によって定期的に割り込みを受けるため、CPU 内の複数の値が、HMI によって同時にすべて更新されることを保証することはできません。 たとえば、HMI 画面に表示されている複数の値のそれぞれが、CPU の異なるスキャンサイクルで取得された値である可能性があります。

PtP (ポイントツーポイント)命令および PROFINET 命令(TSEND_C や TRCV_C など)は、データのバッファを転送しますが、割り込みを受ける可能性があります。 プログラムサイクル OB と割り込み OB の両方で、バッファへの読み取りおよび書き込み操作を禁止することによって、データバッファのデータの一貫性を保証します。 これらの命令用のバッファの値を割り込み OB 内で修正する必要がある場合は、DIS_AIRT 命令を使用して、EN_AIRT 命令が実行されるまで割り込み(割り込み OB や HMI または他の CPU からの通信割り込み)を遅延させます。

注記

DIS_AIRT 命令を使用すると、EN_AIRT 命令が実行されるまで割り込み OB の処理が遅延され、ユーザープログラムの割り込み遅延(イベントの発生から割り込み OB が実行されるまでの時間)に影響を与えます。

5.5 プログラム言語の選択

LAD(ラダーロジック)または FBD(ファンクションブロックダイアグラム)のどちらかの プログラミング言語を選択することができます。

LAD プログラミング言語

LAD はグラフィカルなプログラミング言語です。 表現は回路図がベースです。



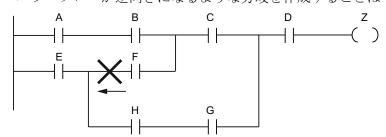
ノーマルクローズとノーマ ルオープンおよびコイルな どの回路図の要素をリンク してネットワークを形成し ます。

複雑な操作を行うロジックを作成するために、分岐を挿入して並列回路のロジックを作成することができます。 並列分岐は下向きに開いているか、母線に直接接続されます。 分岐は上向きに終端します。

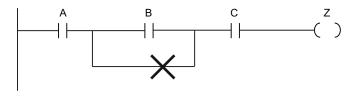
LAD には、math (演算)、timer (タイマー)、counter (カウンタ)、move (ムーブ)など、さまざまな機能のための「ボックス」命令が用意されています。

LAD ネットワークを作成するときは、下記のルールを考慮する必要があります。

- どの LAD ネットワークも coil (コイル)またはボックス命令で終端する必要があります。 ネットワークを Compare (比較)命令や Edge-detection (エッジ検出)(Positive-edge (立ち上がりエッジ)または Negative-edge (立ち下がりエッジ))命令で終端しないようにします。
- パワーフローが逆向きになるような分岐を作成することはできません。



• 短絡を発生させる分岐を作成することはできません。

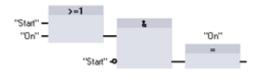


5.5 プログラム言語の選択

ファンクションブロックダイアグラム(FBD)プログラミング言語

FBD も、LAD と同じようにグラフィカルなプログラミング言語です。 論理表現は、ブール代数で使用されているグラフィカルな論理記号がベースになっています。

数学関数などの複雑な関数は、ロジックボックスを併用して直接表現することができます。 複雑な操作を行うロジックを作成するには、ボックス間に並列分岐を挿入します。



「ボックス」命令用の EN および ENO を理解する

LAD および FBD の両方で、「ボックス」命令に「パワーフロー」(EN および ENO)を使用するものがあります。命令によっては(math (演算)や move (ムーブ)など)、EN および ENO 用のパラメータが表示されます。 これらのパラメータはパワーフローに関係があり、そのスキャン中にこの命令を実行するかどうかを決定します。

- EN (Enable In)は LAD および FBD のボックス用ブール入力です。 ボックス命令の 実行が行われるためには、パワーフロー(EN = 1)がこの入力に存在している必要が あります。 LAD ボックスの EN 入力が左側の母線に直接接続されている場合は、ボックスは常に実行されます。
- ENO (Enable Out)は、LAD および FBD のボックス用のブール出力です。 EN 入力においてボックスにパワーフローが存在し、ボックスがその機能を正常に実行した場合、ENO 出力はパワーフロー(ENO = 1)を次の要素に渡します。 ボックス命令の実行でエラーが検出された場合は、エラーを発生したボックス命令でパワーフローが終端されます(ENO = 0)。

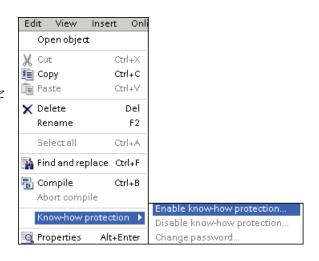
プログラムエデ ィタ	入力/出力	オペランド	データタイ プ
LAD	EN ENO	Power Flow	BOOL
FBD	EN	I、I:P、Q、M、DB、Temp、Power Flow	BOOL
	ENO	Power Flow	BOOL

5.6 コピー保護

コピー保護つまり「ノウハウ」保護によって、作成したプログラム 内の1つ以上のプログラムブロック(OB、FB、FC)への不正なアクセスを防ぐことができます。 プログラムブロックへのアクセスを制限するにはパスワードを作成します。

ブロックに対して「ノウハウ」保護を設定すると、パスワードを入力しないと、そのブロック内のプログラムにアクセスできなくなります。

ブロックをコピーから保護するには、[Edit]メニューで[Know how protection]を選択し、そのブロックにアクセスするためのパスワードを入力します。



パスワード保護によって、そのプログラムブロックへの不正な読み取りや修正を防止することができます。 パスワードを入力しないと、読み取ることのできる情報が、そのプログラムブロックに関する下記の情報に限定されます。

- ブロックタイトル、ブロックコメント、ブロックのプロパティ
- 転送パラメータ(IN、OUT、IN OUT、Return)
- プログラムの呼び出し構造
- クロスリファレンスのグローバルタグ(使用ポイントに関する情報を除く)、ローカルタグは不可

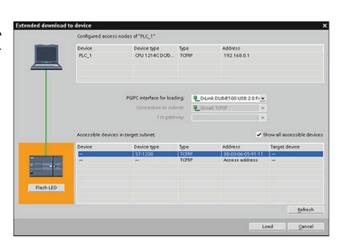
5.7 作成したプログラムの要素をダウンロードする

5.7 作成したプログラムの要素をダウンロードする

プロジェクトの要素をプログラミングデバイスから CPU にダウンロードすることができます。 プロジェクトをダウンロードすると、CPU はユーザープログラム(OB、FB、FC、DB)を不揮発メモリに保持します。

下記のいずれのロケーションからでも、プロジェクトをプログラミングデバイスから CPU にダウンロードすることができます。

- [Project tree]: プログラム要素を右クリックし、コンテキストメニューの [Download]をクリックします。
- [Online]メニュー: [Download to device]をクリックします。
- ツールバー: [Download to device]アイコンをクリック します。



5.8 作成したプログラムの要素をアップロードする

すべてのプログラムブロックをオンライン CPU からオフラインプロジェクトにアップロードすることができますが、デバイス構成とモニタテーブルはアップロードできません。空のプロジェクトにはアップロードできません。アップロードするにはオフライン CPU が必要です。 単一のブロックをアップロードすることはできません。アップロードできるのはプログラム全体です。 アップロードを実行すると、確認応答の後で、アップロードを実行する前に、オフライン CPU が「クリア」されます(すべてのブロックとタグテーブルが削除されます)。 オンライン領域でブロックを編集することはできません。最初にオフライン領域にアップロードし、その領域で修正を行い、PLC にダウンロードして戻す必要があります。

アップロードを実行するには、「プロジェクトツリーでドラッグアンドドロップする」と「比較エディタで同期する」の2つの方法があります。

プロジェクトツリーでドラッグアンドドロップする

- 1. 新規のプロジェクトを作成します。
- 2. アップロード元の CPU に適合する CPU デバイスを追加します。
- 3. CPU ノードを展開して Program blocks (プログラムブロック)フォルダを表示します。
- **4.** プロジェクトツリーの「Online access」のノードを展開し、ネットワークのノードを展開し、「Update accessible devices]をダブルクリックします。
- 5. 使用可能な CPU のリストが表示されたら、目的の CPU のノードを展開します。
- 6. Online access (オンラインアクセス)エリアの Program blocks (プログラムブロック) フォルダをオフラインエリアの Program blocks (プログラムブロック)フォルダにドラッグアンドドロップします。 ドロップするエリアに入るとマウスポインタが「+」に変わります。
- 7. [Upload preview]ダイアログが表示されます。 [Continue]ボックスをオンにして、 [Upload from device]をクリックします。
- **8**. アップロードの実行を許可します。 オフラインエリアに、すべてのプログラムブロック、テクノロジブロック、タグが表示されます。
- 9. デバイス構成はアップロードできないため、「デバイスコンフィグレーション」を使用して、必要な IP アドレスなど、CPU のプロパティの設定および他のデバイスのオフラインプロジェクトへの追加を手動で行います。

オンラインエリアから既存のプログラムの Program blocks(プログラムブロック)エリア にドラッグすることもできます。 とにかく、Program-blocks (プログラムブロック)オ フラインエリアを空にしておく必要はありません。 空にした場合は、既存のプログラムが削除され、オンラインプログラムに置換されます。

比較エディタで同期する

- 1. プロジェクトが含まれているプロジェクトを開きます。
- 2. プロジェクトツリーで、比較するオフライン CPU を選択します。
- 3. 選択したオフライン CPU を右クリックするか、[Tools]メニューから[Compare offline/online]を選択して、「比較」エディタを開きます。
- 4. 比較エディタの Program blocks (プログラムブロック)フォルダの下に相違する部分が表示されます。 操作列のシンボルをクリックします。 プロジェクトをアップロードするには、[Upload from device]を選択します。
- 5. [Synchronize online and offline]ボタンをクリックすると、オンライン CPU からオフライン CPU にプロジェクトがコピーされます。

5.9 プログラムをデバッグして試験する

5.9 プログラムをデバッグして試験する

オンライン CPU によって実行されているユーザープログラムのモニタおよび値の修正を行うには、「モニタテーブル」を使用します。 多様な試験環境をサポートするために、プロジェクト内にさまざまなモニタテーブルを作成および保存することができます。 試運転中やサービスおよびメンテナンスのために、さまざまな試験を繰り返し実施することができます。

モニタテーブルを使用して、ユーザープログラムを実行しながら、CPU のモニタおよび情報のやり取りを行うことができます。 プログラムブロックおよびデータブロックのタグだけではなく、入力および出力(I および Q)、ダイレクトアクセスの入力および出力(I:P および Q:P)、ビットメモリ(M)、データブロック(DB)など、CPU のメモリ領域の値を表示および変更することができます。

モニタテーブルを使用して、STOP モードの CPU の物理出力(Q:P)を有効設定にすることができます。 たとえば、CPU への配線の試験時に、特定の値を出力に割り付けることができます。

また、モニタテーブルを使用して、特定の値をタグに「強制」または設定することもできます。この強制設定の詳細については、「オンラインおよび診断」の章の「CPUに値を強制設定する(ページ 385)」のセクションを参照してください。

プログラミング手順

6.1 基本命令

6.1.1 ビットロジック

LAD 接点

"IN" **— I —**

常時開 "IN" 接点と接点を接続して、独自の組合せ論理回路を作成することができます。 指定した入力ビットがメモリ識別子 I (入力)または Q (出力)を使用している場合、ビット値はプロセスイメージレジスタから読み取られます。制御プロセス内の物理接点信号は PLC の I 端子に配線されています。

CPU は配線されている入力信号をスキャンし、プロセスイメージ入力レジスタの対応する状態値を常時更新しています。

常時閉

 $-\nu$

Iのアドレスに続けて「P:」を使用し (%I3.4:P など)、物理入力の即時読み取りを指定することができます。 即時読み取りでは、ビットデータ値は、プロセスイメージではなく物理入力から直接読み取られます。 即時読み取りではプロセスイメージの更新は行われません。

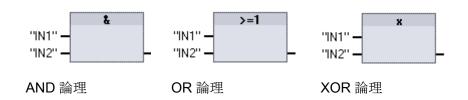
パラメータ	データタイプ	説明
IN	BOOL	割り付け済みビット

- ノーマルオープンは、割り付け済みビットの値が 1 に等しいときに閉じています (ON)。
- ノーマルクローズは割り付け済みビットの値が 0 に等しいときに閉じています(ON)。
- 直列に接続した接点は AND 論理ネットワークを構成します。
- 並列に接続した接点は **OR** 論理ネットワークを構成します。

FBD の AND、OR、 XOR ボックス

FBD プログラミングでは、LAD の論理回路は AND (&)、OR (>=1、排他的 OR (x)ボックスネットワークに変換され、ボックス入力およびボックス出力用のビット値を指定することができます。 また、他のロジックボックスに接続して、独自の論理回路を構成することもできます。 ネットワーク内にボックスを配置した後で、[Favorites]ツールバーまたは命令ツリーから[Insert binary input]ツールをドラッグしてボックスの入力側にドロップし、入力を追加することができます。 ボックス入力コネクタを右クリックして[Insert input]を選択することもできます。

ボックス入力および出力を他のロジックボックスに接続したり、接続されている入力に対してビットアドレスやビットシンボル名を入力することができます。 ボックス命令が実行されると、現在の入力の状態がバイナリボックスロジックに適用され、真の場合はボックス出力が真になります。



パラメータ	データタイプ	説明
IN1、IN2	BOOL	入力ビット

- AND ボックスの出力が TRUE であるためには、すべての入力が TRUE でなければ なりません。
- OR ボックスの出力が TRUE であるためには、どちらか 1 つの入力が TRUE でなければなりません。
- XOR ボックスの出力が TRUE であるためには、どちらか一方の入力が TRUE、も う一方の入力が FALSE でなければなりません。

NOT ロジックインバータ

FBD プログラミングでは、[Favorites]ツールバーまたは命令ツリーから[Negate binary input]をドラッグして入力または出力にドロップすると、そのボックスコネクタにロジ ックインバータが作成されます。



ータ

の AND ボックス

LAD: NOT 接点インバ FBD: 反転論理入力付き FBD: 反転論理入力および出力付き の AND ボックス

LAD での NOT 接点は、パワーフロー入力の論理状態を反転します。

- NOT 接点にパワーフローが存在しない(OFF)場合、パワーフローを出力します(ON)。
- NOT 接点にパワーフローが存在する(ON)場合、パワーフロー出力は存在しません (OFF)

LAD 出力コイル

"OUT" \prec \succ

出力コイル "OUT"

─(/)—

ル

コイル出力命令は出力ビットに値を書き込みます。 指定した出力ビッ トがメモリ識別子 Q を使用している場合、CPU は、プロセスイメージ レジスタ内のその出力ビットをオンまたはオフにセットして、指定し たビットがパワーフローステータスに等しくなるように設定します。 制御アクチュエータへの出力信号は S7-1200 の Q 端子に配線されてい ます。RUN モードでは、CPU システムは、常時入力信号のスキャン 反転出力コイ を行い、入力の状態をプログラムロジックに基づいて処理し、プロセ スイメージ出力レジスタに新しい出力状態の値を設定することで応答 します。 プログラム実行サイクルを実行するたびに、CPU システム は、プロセスイメージレジスタに格納されている新しい出力状態応答 を配線先の出力端子に転送します。

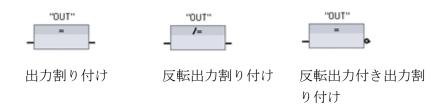
Qアドレスに続けて「P:」を使用し (%Q3.4:P など)、物理出力への即時書き込みを指定することができます。 即時書き込みでは、ビットデータ値がプロセスイメージ出力に書き込まれるほか、物理出力にも直接書き込まれます。

パラメータ	データタイプ	説明
OUT	BOOL	割り付け済みビット

- 出力コイルを通るパワーフローが存在する場合、出力ビットが 1 にセットされます。
- 出力コイルを通るパワーフローが存在しない場合、出力ビットが 0 にセットされます。
- 反転出力コイルを通るパワーフローが存在する場合、出力ビットが 0 にセットされます。
- 反転出力コイルを通るパワーフローが存在しない場合、出力ビットが 1 にセットされます。

FBD 出力割り付けボックス

FBD プログラミングでは、LAD コイルは割り付け(=and/=) ボックスに変換され、そのボックス出力のビットアドレスを指定することができます。 ボックスの入力および出力を他のボックスロジックに接続したり、ビットアドレスを入力することができます。



パラメータ	データタイプ	説明
OUT	BOOL	割り付け済みビット

- 出力ボックス入力が 1 の場合、出力ビットが 1 にセットされます。
- 出力ボックス入力が 0 の場合、出力ビットが 0 にセットされます。
- 反転出力ボックス入力が 1 の場合、出力ビットが 0 にセットされます。
- 反転出力ボックス入力が 0 の場合、出力ビットが 1 にセットされます。

6.1.1.1 セットおよびリセット命令

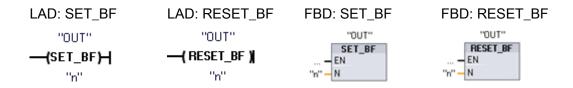
Sおよび R: 1 ビットをセットおよびリセットする

- S (Set)をアクティブにすると、OUT アドレスのデータ値が 1 にセットされます。S をアクティブにしなかった場合は OUT は変更されません。
- R (Reset) をアクティブにすると、OUT アドレスのデータ値が 0 にセットされます。 R をアクティブにしなかった場合は OUT は変更されません。
- これらの命令はネットワーク内のどこにでも配置することができます。

LAD: SET	LAD: RESET	FBD: SET	FBD: RESET
"OUT"	"OUT"	"OUT"	"OUT"
—(s)—	—(R)—	"IN" — S	'1N'' — R

パラメータ	データタイプ	説明
IN (または接点/タグへの接続ロジ	BOOL	モニタするビットのロケーション
ック)		
OUT	BOOL	セットまたはリセットするビット
		のロケーション

SET_BF および RESET_BF: ビットフィールドをセットおよびリセットする



パラメータ	データタイプ	説明
n	Constant	書き込むビット数
OUT	ブール配列の要素	セットおよびリセットするビットフィールド の開始要素
		例 #MyArray[3]

- SET_BF をアクティブにすると、アドレス OUT から始まる n ビットに 1 の値が割り付けられます。 SET_BF をアクティブにしなかった場合は OUT は変化しません。
- RESET_BF は、アドレス OUT から始まる n ビットに 0 の値を書き込みます。 RESET_BF をアクティブにしなかった場合は OUT は変化しません。
- これらの命令は、分岐の右端に配置する必要があります。

RS および SR: セット優先およびリセット優先ビットのラッチ



- R1

RS は、セット優先のセット優先ラッチです。 セット(S1)信号とリセット(R)信号の両方が真の場合、出力アドレス OUT が 1 になります。

SR は、リセット優先のリセット優先ラッチです。 セット(S)信号とリセット(R1)信号の両方が真の場合、出力アドレス OUT が 0 になります。

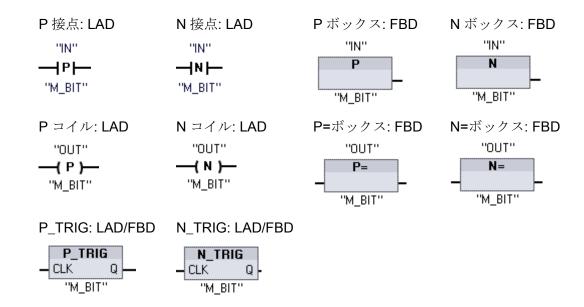
OUT パラメータは、セットまたはリセットするビットアドレスを指定します。 オプションの OUT 出力 Q は、OUT アドレスの信号の状態を反映します。

パラメータ	データタイプ	説明
S、S1	BOOL	入力をセットする、S1 は優先であることを表す
R、R1	BOOL	入力をリセットする、R1 は優先である ことを表す
OUT	BOOL	割り付けられたビット出力「OUT」
Q	BOOL	「OUT」ビットの状態に従う

命令	S1	R	「OUT」ビット
RS	0	0	前の状態
	0	1	0
	1	0	1
	1	1	1
	S	R1	
SR	0	0	前の状態
	0	1	0
	1	0	1
	1	1	0

6.1.1.2 ポジティブおよびネガティブエッジ命令

ポジティブエッジおよびネガティブエッジ命令



パラメータ	データタ	説明
	17	
M_BIT	BOOL	前の入力の状態が保持されているメモリビット
IN	BOOL	エッジを検出する入力ビット
OUT	BOOL	エッジが検出されたことを示す出力ビット
CLK	BOOL	パワーフローまたはエッジを検出する入力ビット
Q	BOOL	エッジが検出されたことを示す出力ビット

P接点: この接点の状態は、割り付けられている IN ビット上でポジティブエッジ

LAD (OFF から ON へ) が検出されると TRUE になります。 接点の論理状態は、パワーフローの入力状態と組み合わされてパワーフローの出力状態を設定します。 P接点は、分岐の終りを除き、ネットワーク内のどこ

にでも配置できます。

N接点: この接点の状態は、割り付けられている IN ビット上でネガティブエッジ

LAD (ON から OFF へ)が検出されると TRUE になります。 接点の論理状態は、パワーフローの入力状態と組み合わされてパワーフローの出力状態を設定します。 N 接点は、分岐の終りを除き、ネットワーク内のどこにでも配置できます。

Pボックス: この出力ロジックの状態は、割り付けられている IN ビット上でポジティ FBD ブエッジ(OFF から ON へ)が検出されると TRUE になります。 Pボック スは、分岐の始まりにのみ配置することができます。

N ボックス: この出力ロジックの状態は、割り付けられている IN ビット上でネガティ FBD ブエッジ(ON から OFF \sim)が検出されると TRUE になります。 N ボックスは、分岐の始まりにのみ配置することができます。

P コイル: コイルに入力されるパワーフロー上でポジティブエッジ(OFF から ON LAD へ)が検出されると、割り付けられているビット「OUT」が TRUE になります。 パワーフローの入力状態は、パワーフローの出力状態として常にコイルを通過します。 P コイルは、ネットワーク内のどこにでも配置できます。

N コイル: コイルに入力されるパワーフロー上でネガティブエッジ(ON から OFF LAD へ)が検出されると、割り付けられているビット「OUT」が TRUE になります。 パワーフローの入力状態は、パワーフローの出力状態として常にコイルを通過します。 N コイルは、ネットワーク内のどこにでも配置できます。

P=ボック ボックスが分岐の始点に配置されている場合に、ボックス入力接続の論 ス: 理状態または入力ビット割り付け上でポジティブエッジ (OFF から ON トリング では、割り付けられているビット「OUT」が TRUE になります。 入力の論理状態は、出力の論理状態として常にボックスを通過します。 P=ボックスは分岐内のどこにでも配置できます。

N=ボック ボックスが分岐の始点に配置されている場合に、ボックス入力接続の論 理状態または入力ビット割り付け上でネガティブエッジ (ON から OFF FBD へ) が検出されると、割り付けられているビット「OUT」が TRUE になります。 入力の論理状態は、出力の論理状態として常にボックスを通過します。 N=ボックスは分岐内のどこにでも配置できます。

P_TRIG: CLK 入力状態(FBD)または CLK パワーフロー入力(LAD)上でポジティブ

LAD/FBD エッジ(OFF から ON \sim)が検出されると、Q 出力パワーフローまたは論

理状態が TRUE になります。 LAD では、 P_TRIG 命令をネットワークの 始点または終点に配置することはできません。 FBD では、分岐の終りを

除き、P_TRIG 命令をどこにでも配置できます。

N_TRIG CLK 入力状態(FBD)または CLK パワーフロー入力(LAD)上でネガティブ

(LAD/FBD) エッジ(ON から OFF \sim)が検出されると、Q 出力パワーフローまたは論

理状態が TRUE になります。 LAD では、N_TRIG 命令をネットワークの 始点または終点に配置することはできません。 FBD では、分岐の終りを

除き、P_TRIG 命令をどこにでも配置できます。

すべてのエッジ命令は、モニタする入力信号の前の状態を保持するためにメモリビット (M_BIT)を使用します。 エッジの検出は、入力の状態をメモリビットに保持されている 状態と比較することによって行われます。 状態が所定の方向に変化した場合は、出力に TRUE を書き込むことによってエッジの検出が報告されます。 そうでない場合は、出力に FALSE が書き込まれます。

注記

エッジ命令は、最初の実行を含め、実行されるたびに入力値とメモリビットの値を評価 します。 プログラム設計で、最初のスキャンでエッジ検出を許可するため、または禁 止するために、入力とメモリビットの初期状態を考慮する必要があります。

メモリビットは次の実行まで保持する必要があるため、それぞれのエッジ命令に固有のビットを使用し、そのビットをプログラム内の他の場所で使用しないようにする必要があります。 また、テンポラリメモリや、I/O の更新などの他のシステムファンクションによって影響を受ける可能性のあるメモリを避ける必要があります。 M_BIT メモリの割り付けには、M、グローバル DB、またはスタティックメモリ(インスタンス DB 内)のみを使用するようにします。

6.1.2 タイマ

タイマ命令は、プログラムによる遅延を作成するために使用します。

- TP: パルスタイマは設定した幅のパルスを生成します。
- TON: オンディレイタイマ出力 Q は、設定した遅延時間経過後に ON にセットされます。
- TOF: オフディレイタイマ出力 Q は、設定した遅延時間経過後に OFF にセットされます。
- TONR: 保持型オンディレイタイマ出力は、設定した遅延時間経過後に ON にセットされます。 経過した時間は、R 入力を使用して経過時間をリセットするまで、複数の時間にわたって累積されます。
- RT: 指定したタイマのインスタンスデータブロックに格納されている時間データを クリアしてタイマをリセットします。

それぞれのタイマは、データブロック内に格納された構造体を使用して、タイマデータを保持します。 エディタでタイマ命令を配置するときに、データブロックを割り付けます。

ファンクションブロックにタイマ命令を配置するとき、マルチインスタンスデータブロックオプションを選択することができます。そうすれば、タイマ構造体の名前を変えてデータ構造体を個別に用意することができますが、タイマデータは単一のデータブロック内に格納され、タイマごとにそれぞれのデータブロックを用意する必要がありません。これにより、タイマの処理に必要な処理時間とデータストレージを節約することができます。 共有するマルチインスタンスデータブロック内のタイマデータ構造体間での相互作用はありません。



TP、TON、TOF タイマには同じ入力および出力パラメータが存在します。



TONR タイマには、さらにリセット入力パラメータ R が存在します。

タイマデータブロックの名前として、作成したプロセス内でのこの タイマの目的を表す独自の「タイマ名」を作成します。

タイマ名 ----[RT]----

RT命令は、指定したタイマのタイマデータをリセットします。

パラメータ	データタイプ	説明
IN	BOOL	タイマ入力
R	BOOL	TONR 経過時間をゼロにリセットする
PT	BOOL	タイマ設定値入力
Q	BOOL	タイマ出力
ET	Time	経過時間値出力
Timer data block	DB	RT命令でリセットするタイマを指定する

パラメータ IN はタイマを開始および停止します。

- パラメータ IN が 0 から 1 に遷移すると、タイマ TP、TON、TONR が開始します。
- パラメータ IN が 1 から 0 に遷移すると、タイマ TOF が開始します。

PT パラメータと IN パラメータの値の変化の影響を下の表に示します。

タイマ	PT パラメータと IN パラメータの値の変化
TP	• このタイマの実行中に PT が変化しても、影響はありません。
	• このタイマの実行中に IN が変化しても、影響はありません。
TON	• このタイマの実行中に PT が変化しても、影響はありません。
	• このタイマの実行中に IN が FALSE に変化すると、タイマはリセット され停止します。
TOF	• このタイマの実行中に PT が変化しても、影響はありません。
	• このタイマの実行中に IN が TRUE に変化すると、タイマはリセットされ停止します。
TONR	• このタイマの実行中に PT が変化しても影響はありませんが、タイマの 復帰時に影響を与えます。
	• このタイマの実行中に IN が FALSE に変化すると、タイマは停止しますがリセットは行われません。 IN が再び TRUE に変化すると、タイマは累積された時間値から計時を再開します。

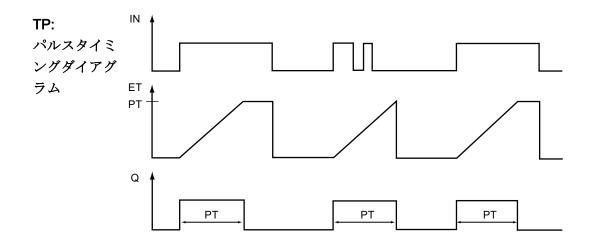
TIME 値

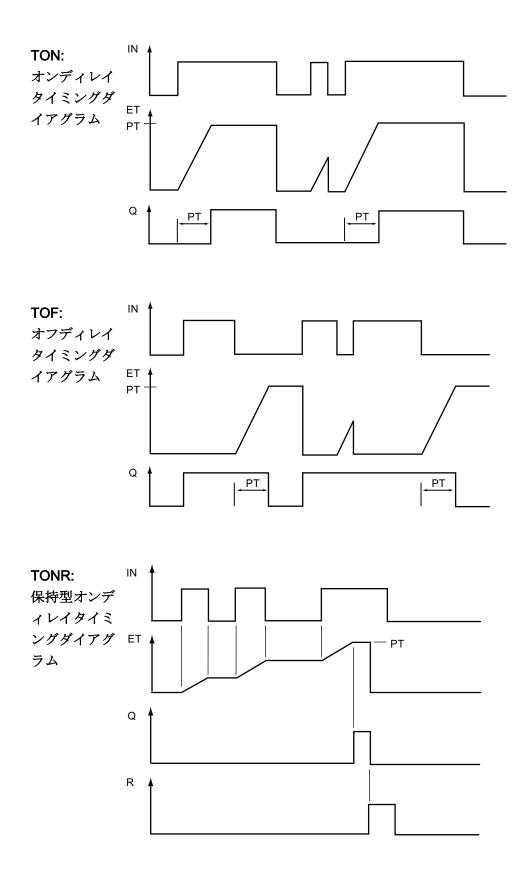
PT (タイマ設定値)および ET (経過時間)の値は、ミリ秒単位の時間を表す符号付き倍精度整数としてメモリに格納されます。 TIME データには識別子 T#が使用され、単純な時間単位(T#200ms など)または複合時間単位(T#2s_200ms など)として入力することができます。

データ タイプ	サイズ	有効な値の範囲
TIME	32 ビット	T#-24d_20h_31m_23s_648ms~T#24d_20h_31m_23s_647ms
	(格納値)	-2,147,483,648 ms~+2,147,483,647 ms

注記

上に示した TIME データタイプの負の値の範囲はタイマ命令には使用できません。 負の PT (タイマ設定)値は、タイマ命令の実行時にゼロにセットされます。 ET (経過時間) は常に正の値です。





6.1.3 カウンタ

6.1.3.1 カウンタ

内部プログラムイベントおよび外部プロセスイベントのカウントを行うにはカウンタ命令を使用します。

- CTU はカウントアップカウンタです。
- CTD はカウントダウンカウンタです。
- CTUD はカウントアップダウンカウンタです。

それぞれのカウンタは、データブロック内に格納された構造体を使用して、カウンタデータを保持します。エディタでカウンタ命令を配置するときに、データブロックを割り付けます。これらの命令はソフトウェアカウンタを使用しているため、配置されている OB の実行速度によって最大カウント速度が制限されます。カウンタ命令が呼び出される OB は、CU または CD 入力のすべての遷移を検出するのに十分な頻度で実行される必要があります。高速度でのカウント操作については、CTRL_HSC 命令を参照してください。

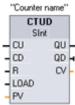
ファンクションブロックにカウンタ命令を配置するとき、マルチインスタンスデータブロックオプションを選択することができます。そうすれば、カウンタ構造体の名前を変えてデータ構造体を個別に用意することができますが、カウンタデータは単一のデータブロック内に格納され、カウンタごとにそれぞれのデータブロックを用意する必要がありません。これにより、カウンタの処理に必要な処理時間とデータストレージを節約することができます。 共有するマルチインスタンスデータブロック内のカウンタデータ構造体間での相互作用はありません。



ボックス名の下のドロップダウンリストでカウンタ値のデータタイプ を選択します。



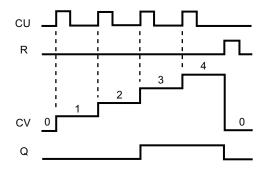
カウンタデータブロックの名前として、作成したプロセス内でのこの カウンタの目的を表す独自の「カウンタ名」を作成します。



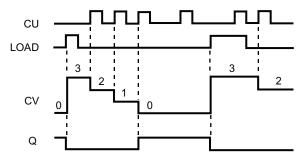
パラメータ	データタイプ	説明
CU, CD	BOOL	1カウントずつ増数または減数します
R (CTU、CTUD)	BOOL	カウンタ値をゼロにリセットします
LOAD (CTD、 CTUD)	BOOL	カウンタ設定値のロード制御
PV	SInt, Int, DInt, USInt, UInt, UDInt	カウンタ設定値
Q, QU	BOOL	CV >= PV の場合に真
QD	BOOL	CV <= 0 の場合に真
CV	SInt, Int, DInt, USInt, UInt, UDInt	現在のカウンタ値

カウンタ値の範囲は選択したデータタイプに依存します。 カウンタ値が符号なし整数型の場合は、0までカウントダウンまたは範囲の限度までカウントアップすることができます。 カウンタ値が符号付き整数型の場合は、負の整数の限度までカウントダウンまたは正の整数の限度までカウントアップすることができます。

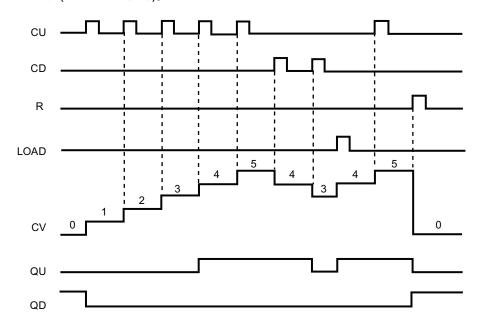
CTU: CTU は、パラメータ CU が 0 から 1 に変化すると 1 カウント加算します。パラメータ CV (現在のカウンタ値)がパラメータ PV (カウンタ設定値)以上になった場合は、カウンタ出力 Q が 1 になります。リセットパラメータ R の値が 0 から 1 に変化した場合は、現在のカウンタ値が 0 にリセットされます。符号なし整数のカウンタ値の CTU のタイミングダイアグラムを次の図に示します(PV = 3 の場合)。



CTD: パラメータ CD の値が 0 から 1 に変化すると CTD は 1 カウント減算します。パラメータ CV の値(現在のデータ値)が 0 以下になるとカウンタの出力パラメータ Q が 1 になります。パラメータ LOAD の値が 0 から 1 に変化すると、パラメータ PV (カウンタ設定値)の値を新しい CV(現在のデータ値)としてカウンタにロードされます。 符号なし整数のカウンタ値を使用した CTD のタイミングダイアグラムを下の表に示します (PV = 3 の場合)。



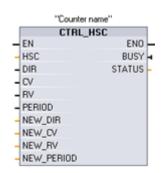
CTUD: CTUD は、カウントアップ(CU)入力またはカウントダウン(CD)入力が 0 から 1 に変化すると 1 カウント加算または減算します。 パラメータ CV(現在のデータ値)の値がパラメータ PV (カウンタ設定値)の値以上になると、カウンタの出力 QU が 1 になります。パラメータ CV の値がゼロ以下になった場合は、カウンタの出力 QD が 1 になります。パラメータ LOAD の値が 0 から 1 に変化すると、パラメータ PV (カウンタ設定値)の値が新しい CV(現在のデータ値)としてカウンタにロードされます。 リセットパラメータ R の値が 0 から 1 に変化すると、現在のカウンタ値が 0 にリセットされます。 符号なし整数のカウンタ値を使用した CTUD のタイミングダイアグラムを下の表に示します(PV = 4 の場合)。



6.1.3.2 CTRL_HSC命令

CTRL_HSC命令は、OBの実行速度よりも高速に発生するイベントのカウントに使用する高速カウンタを制御します。 CTU、CTD、およびCTUDカウンタ命令のカウント速度は、配置されているOBの実行速度によって制限されます。 HSCの最大入力クロック周波数については、技術仕様 (ページ 398)を参照してください。

一般に、モーション制御のシャフトエンコーダによって生成されるパルスのカウントに 高速カウンタを使用します。



CTRL_HSC 命令のそれぞれが、データブロック内に格納されている構造体を使用してデータを保持します。 エディタで CTRL_HSC 命令を配置するときに、データブロックを割り付けます。

カウンタデータブロックの名前として、作成したプロセス内でのこのカウンタの目的を表す独自の「カウンタ名」を作成します。

パラメータ	パラメータ タイプ	データタイ プ	説明
HSC	IN	HW_HSC	HSC 識別子
DIR	IN	BOOL	1 = 方向の変更を要求する
CV	IN	BOOL	1 = 新しいカウンタ値をセットするよう に要求する
RV	IN	BOOL	1 = 新しい基準値をセットするように要求する
PERIOD	IN	BOOL	1 = 新しい周波数計測時間の値をセット するように要求する (周波数計測モードのみ)
NEW_DIR	IN	INT	新しい方向: 1= 前進 -1= 後退
NEW_CV	IN	DINT	新しいカウンタ値
NEW_RV	IN	DINT	新しい基準値

パラメータ	パラメータ タイプ	データタイ プ	説明
NEW_PERIOD	IN	INT	秒単位の新しい周波数計測時間 .01、.1、または 1 (周波数計測モードのみ)
BUSY	OUT	BOOL	ファンクションビジー
STATUS	OUT	WORD	実行条件コード

プログラム内で高速カウンタを使用するには、プロジェクト設定の PLC のデバイスコンフィグレーションで、高速カウンタを事前に設定しておく必要があります。 HSC のデバイスコンフィグレーションの設定で、カウンタモード、I/O 接続、割り込みの割り付け、高速カウンタとして使用するのかパルス周波数計測デバイスとして使用するのかを選択します。 高速カウンタの操作は、プログラム制御を使用して行うことも、プログラム制御を使用せずに行うこともできます。

高速カウンタのパラメータの多くは、プロジェクトのデバイスコンフィグレーションでのみセットされます。 プロジェクトのデバイスコンフィグレーションによって初期化される高速カウンタパラメータもありますが、プログラム制御によって後から変更することができます。

CTRL_HSC 命令のパラメータを使用して、カウントプロセスのプログラム制御を行うことができます。

- カウント方向を NEW_DIR 値にセットする
- 現在のカウント値を NEW CV 値にセットする
- 基準値を NEW RV にセットする
- 周波数計測時間を NEW_PERIOD にセットする(周波数計測モード)

CTRL_HSC の実行時に下記のフラグが 1 にセットされた場合は、対応する新しい値がカウンタにロードされます。 複数の要求が(同時に複数のフラグがセットされたとき)、CTRL HSC 命令の 1 回の実行で処理されます。

- DIR = 1 は新しい NEW DIR 値をロードするための要求で、DIR = 0 は変化なし
- CV = 1 は新しい NEW_CV 値をロードするための要求で、CV = 0 は変化なし
- RV = 1 は新しい NEW_RV 値をロードするための要求で、RV = 0 は変化なし
- PERIOD = 1 は新しい NEW_PERIOD 値をロードするための要求で、PERIOD = 0 は変化なし

一般に、CTRL_HSC 命令は、カウンタハードウェア割り込みイベントがトリガされた時に実行するハードウェア割り込み OB 内に配置します。 たとえば、CV=RV イベントがカウンタ割り込みをトリガした場合、ハードウェア割り込み OB プログラムブロックが CTRL_HSC 命令を実行し、NEW_RV 値をロードすることによって基準値を変更することができます。

CTRL_HSC のパラメータに現在のカウンタ値を使用することはできません。 現在のカウンタ値が格納されているプロセスイメージアドレスは、高速カウンタのハードウェアコンフィグレーション中に割り付けられます。 プログラムロジックを使用してカウンタ値を直接読み取った場合、プログラムに返された値がカウンタを読み取った時点での正しいカウンタ値になります。 高速カウンタのイベントは続行されます。 したがって、プログラムで、このカウンタ値を使用して処理を完了する前に、実際のカウンタ値が変化している可能性があります。

CTRL HSC パラメータの詳細

- パラメータ値の更新が要求されなかった場合は、対応する入力値は無視されます。
- DIR パラメータは、設定されているカウントの方向が、[User program (internal direction control)]に設定されている場合にのみ有効です。 HSC のデバイスコンフィグレーションで、このパラメータの使用方法を指定します。
- CPU またはシグナルボード上の S7-1200 HSC の場合、BUSY パラメータの値は常 に 0 です。

条件コード: エラーが発生した場合は ENO が 0 にセットされ、STATUS 出力に状態コードが格納されます。

STATUS 値 (W#16#)	説明	
0	エラーは発生していません。	
80A1	HSC 識別子は HSC のアドレス指定ではありません	
80B1	NEW_DIR の値が不正です	
80B2	NEW_CV の値が不正です	
80B3	NEW_RV の値が不正です	
80B4	NEW_PERIOD の値が不正です	

6.1.3.3 高速カウンタの操作

高速カウンタはインクリメンタルシャフトエンコーダ用の入力として使用することができます。 このシャフトエンコーダは、1回転あたりのカウント数を指定でき、1回転ごとにリセットパルスを発生させることができます。 シャフトエンコーダからのクロックおよびリセットパルスを HSC への入力として使用します。

HSC にはいくつかの設定値の初期値がロードされ、現在のカウンタ値が現在の設定値以下の時間はアクティブになります。 HSC は、現在のカウンタ値が設定値と等しくなったとき、リセットが発生したとき、および方向が変化したときに、割り込みを発生させます。

現在のカウンタ値が設定値に等しくなったことによって発生する割り込みのたびに、新 しい設定値がロードされ、次の出力状態がセットされます。 リセットイベントが発生 した場合は、最初の設定値と最初の出力の状態がセットされ、サイクルが反復されます。

割り込みの発生頻度は HSC のカウンタ速度よりもはるかに低いため、高速カウンタの 正確な制御を実装しても、CPU のスキャンサイクルに与える影響を比較的小さく抑え ることができます。 割り込みを付加する方法によって、複数の新しい設定値のロード を別の割り込みルーチンで実行することができ、状態制御を簡単に行うことができます (また、すべての割り込みイベントを 1 つの割り込みルーチンで処理することもできます)。

HSC の機能を選択する

すべての HSC は、同じカウンタモードの動作では同じように機能します。 HSC には 4 つの基本的なタイプがあります。

- 内部方向制御式単相カウンタ
- 外部方向制御式単相カウンタ
- 2クロック入力式2相カウンタ
- A/B 相直角位相カウンタ

それぞれの HSC をリセット入力付き、または無しで使用することができます。 リセット入力をアクティブにすると(制限事項については下表参照)、現在値がクリアされ、リセット入力を解除するまでクリアした状態が保持されます。

● 周波数ファンクション: HSC のモードによっては、現在のパルス数ではなく周波数をカウントするように構成できます(カウンタ型)。 周波数計測時間として、 0.01、 0.1、1.0 秒の 3 種類が用意されています。

周波数計測時間は、HSC での新しい周波数値の計算頻度を決定します。 報告される 周波数は、直前の計測時間のカウンタ値の合計によって決定される平均値です。 周 波数が高速に変化している場合は、その計測時間に発生した最高と最低の周波数の 中間値になります。 周波数計測時間の設定に関係なく、周波数は常にヘルツ単位(1 秒間あたりのパルスの数)で報告されます。

● カウンタのモードおよび入力 **HSC** に関連のあるクロック、方向制御、リセット機能用の入力を下の表に示します。

2つの異なる機能に同じ入力を使用することはできませんが、その HSC の現在のモードで使用していない入力を別の目的に使用することができます。 たとえば、HSC1 が内部入力を使用し、外部リセット(I0.3)を使用しないモードのとき、I0.3 をHSC2 のエッジ割り込みに使用することができます。

説明			デフォルトの入力割り付け		け	ファンクション
HSC	HSC1	内蔵 またはシグナルボー ド またはモニタ PTO 01		I0.1 I4.1 PTO 0 Direction	10.3 14.3 -	
	HSC:	内蔵 またはシグナルボー		I0.3 I4.3 PTO 1 Direction	10.1 14.1 -	
	HSC3 ²	内蔵	10.4	10.5	10.7	
	HSC4 ³	内蔵	10.6	10.7	10.5	
	HSC5⁴	内蔵 またはシグナルボー ド		l1.1 l4.1	I1.2 I4.3	
	HSC6 ⁴	内蔵 またはシグナルボー ド		11.4 14.3	I1.5 I4.1	
モード	内部方向	制御式単相カウンタ	クロック	-	-	カウンタ値または 周波数
					RESET	カウント
	外部方向	制御式単相カウンタ	クロック	方向	-	カウンタ値または 周波数
					RESET	カウント
	2 クロツ:	ク入力式 2 相カウン	Clock up	Clock down	-	カウンタ値または 周波数
					RESET	カウント

説明	デフォルトの入力割り付け			ファンクション
A/B 相直角位相カウンタ	Phase A	Phase B	-	カウンタ値または 周波数
			Phase Z	カウント
パルストレイン出力モニタ (PTO) ¹	クロック	方向	-	カウント

- 1 パルストレイン出力モニタでは常にクロックと方向が使用されます。 対応する PTO がパルスのみに設定されている場合は、方向出力は通常はポジティブカウンティング用にセットされます。
- ² CPU 1211C は、内部入力が 6 チャンネルしかサポートされていないため、リセット入力付きの HSC3 を使用することはできません。
- 3 CPU 1211C は、内部入力が 6 チャンネルしかサポートされていないため、HSC4 を使用することはできません
- 4 CPU 1211C および CPU 1212C では、シグナルボードを追加した場合にのみ HSC5 および HSC6 を使用することができます。

HSC の現在値にアクセスする

CPU は、HSC の現在値を入力(I)アドレスに格納します。 各 HSC の現在値に割り付けられるデフォルトアドレスを下の表に示します。 デバイスコンフィグレーションで CPU のプロパティを変更することによって、現在値の I アドレスを変更することができます。

高速カウンタ	データタイプ	デフォルトのアドレス
HSC1	DINT	ID1000
HSC2	DINT	ID1004
HSC3	DINT	ID1008
HSC4	DINT	ID1012
HSC5	DINT	ID1016
HSC6	DINT	ID1020

HSC デバイスに割り付けられるデジタル I/O ポイントの強制はできません

高速カウンタデバイスで使用されるデジタル I/O ポイントは、デバイスコンフィグレーション時に割り付けられます。 これらのデバイスにデジタル I/O ポイントのアドレスが割り付けられると、割り付けられた I/O ポイントのアドレスをモニタテーブルの強制ファンクションで変更することはできません。

6.1.3.4 HSCのコンフィグレーション

この CPU では、最大 6 つの高速カウンタを構成することができます。 個別の HSC のパラメータを設定するには、CPU の「プロパティ」を編集します。

CPU の「プロパティ」を編集することによって、 高速カウンタのパラメータを設定します。

HSC を有効設定にした後で、カウンタ機能、初期値、リセットオプション、割り込みイベントなど、他のパラメータを設定します。

HSC のコンフィグレーションが終了したら、ユーザープログラム内で CTRL_HSC 命令を使用して、HSC の動作を制御します。

General

- General
- ▶ PROFINET interface
- ▶ DI14/DO10
- ▶ Al2

High speed counters (HSC)

- ▶ High speed counters (HSC)1
- ▶ High speed counters (HSC)2
- ▶ High speed counters (HSC)3
- High speed counters (HSC)4
- High speed counters (HSC)5High speed counters (HSC)6
- ▶ Pulse generators (PTO/PWM)

Startup

Time of day

Protection

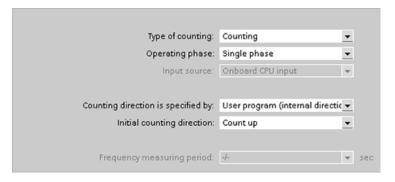
System and clock memory

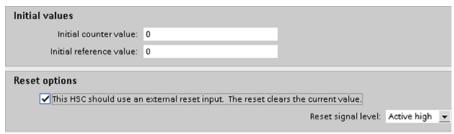
Cycle time

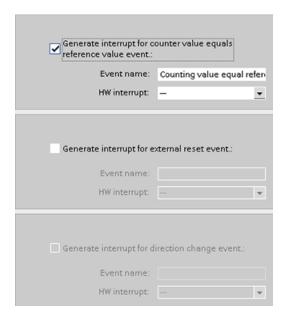
Communication load

Overview of I/O addresses

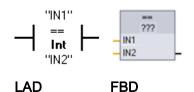








6.1.4 比較



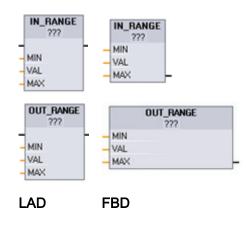
同じデータタイプの2つの値を比較するには比較命令を使用します。 LAD の接点比較が TRUE の場合は、その接点が ON になります。 FBD のボックス比較が TRUE の場合は、そのボックスの出力が OFF になります。

プログラムエディタで命令をクリックし、ドロップダウンメニューで比較タイプとデータタイプを選択することができます。

関係	次の場合に出力状態が ON になります
==	IN1とIN2が等しい
<>	IN1とIN2が等しくない
>=	IN1がIN2に等しいか、それ以上
<=	IN1はIN2以下
>	IN1 が IN2 より大きい
<	IN1 が IN2 より小さい

パラメータ	データタイプ	説明
IN1, IN2	SInt, Int, DInt, USInt, UInt, UDInt, Real,	比較する 2 つの値
	LReal, String, Char, Time, DTL, Constant	

範囲内および範囲外命令



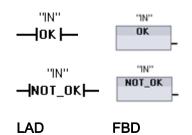
入力値が指定した範囲内であるか範囲外であるかを調べるには、IN_RANGE および OUT_RANGE 命令を使用します。 比較が TRUE の場合は、そのボックスの出力が TRUE になります。

入力パラメータ MIN、VAL、および MAX は、 同じデータタイプでなければなりません。 プログラムエディタで命令をクリックし、ドロ ップダウンメニューでデータタイプを選択する ことができます。

関係	次の場合に比較が TRUE になります
IN_RANGE	MIN <= VAL <= MAX
OUT_RANGE	VAL < MIN または VAL > MAX

パラメータ	データタイプ	説明
MIN, VAL,	SInt, Int, DInt, USInt, UInt, UDInt, Real,	コンパレータ入力
MAX	Constant	

OK および Not OK 命令



入力データ参照が IEEE 754 に基づく有効な実数であるかどうかを調べるには、OK および NOT_OK 命令を使用します。LAD 接点が TRUE のとき、その接点がアクティブになって ON の信号を渡します。 FBD ボックスがTRUE のとき、そのボックスの出力は TRUE です。

Real または LReal 値は、+/- INF (無限大)、NaN (数値でない)、または非正規化値の場合は無効です。 非正規化値とは限りなくゼロに近い値のことです。 CPU での計算時には、非正規化値の代わりにゼロが使用されます。

命令	次の場合は Real の数値テストは TRUE です	
ОК	入力値が有効な Real の数である	
NOT_OK	入力値が有効な Real の数ではない	

パラメータ	データタイプ	説明
IN	Real, LReal	入力データ

6.1.5 算術演算命令

加算、減算、乗算、除算命令



基本的な算術演算をプログラムするには Math ボックス命令を使用します。

- ADD: 加算(IN1 + IN2 = OUT)
- SUB: 減算(IN1 IN2 = OUT)
- MUL: 乗算(IN1 * IN2 = OUT)
- DIV: 除算(IN1 / IN2 = OUT)
 整数の除算では、整数の出力を生成するために、小数点以下の部分は切り捨てられます。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

基本的算術演算命令のパラメータの IN1、IN2、および OUT は、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
IN1, IN2	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Constant	算術演算入力
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	算術演算出力

算術演算命令を有効設定(EN = 1)にすると、入力値(IN1 および IN2)に対して指定した演算を実行し、その結果を出力パラメータ(OUT)で指定したメモリアドレスに格納します。 演算が正常に終了すると、命令は ENO を 1 にセットします。

ENO のス テータス	説明
1	エラーは発生していません。
0	算術演算の結果値が、選択したデータタイプの有効な範囲を超えていま す。 有効なサイズに適合する結果の最小部が返されます。
0	0 での除算(IN2 = 0): 結果が未定義のためゼロが返されます。

ENO のス テータス	説明
0	Real/LReal: 入力値の 1 つが NaN (数値でない)の場合は NaN が返されます。
0	ADD Real/LReal: 両方の IN の値が符号の異なる INF の場合は不正な演算であり NaN が返されます。
0	SUB Real/LReal: 両方の IN の値が同符号の INF の場合は不正な演算であり NaN が返されます。
0	MUL Real/LReal: 一方の IN の値がゼロで他方が INF の場合は不正な演算であり NaN が返されます。
0	DIV Real/LReal: 両方の IN の値がゼロまたは INF の場合は不正な演算であり NaN が返されます。

6.1.5.1 MOD命令



MOD (剰余算)命令は、IN1 MOD IN2 演算を行うために使用します。 演算 IN1 MOD IN2 = IN1 - (IN1 / IN2)の結果はパラメータ OUT に格納 されます。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

IN1、IN2、OUT の各パラメータは、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
IN1 および IN2	Int, DInt, USInt, UInt, UDInt, Constant	剰余算入力
OUT	Int, DInt, USInt, UInt, UDInt	剰余算出力

ENO のス テータス	説明
1	エラーは発生していません。
0	値 IN2 = 0、OUT には値ゼロが代入される

NEG 命令



NEG (2 の補数)命令は、パラメータ IN の値の符号を反転し、その結果をパラメータ OUT に格納するために使用します。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

IN および OUT パラメータは、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
IN	SInt, Int, Dint, Real, LReal, Constant	数値演算入力
OUT	SInt, Int, DInt, Real, LReal	数値演算出力

ENO のステ ータス	説明
1	エラーは発生していません。
0	結果の値が、選択したデータタイプの有効な範囲を超えています。 SInt の場合の例: NEG (-128)の結果は+128 になり、データタイプ maximum の範囲を超えています。

インクリメント命令とデクリメント命令





次の場合に INC 命令および DEC 命令を使用します。

- 符号付きまたは符号なし整数値の値を増加する
 INC (インクリメント): パラメータ IN/OUT の値 +1 = パラメータ
 IN/OUT の値
- 符号付きまたは符号なし整数値を減少させる
 DEC (デクリメント): パラメータ IN/OUT の値 1 = パラメータ IN/OUT の値

パラメータ	データタイプ	説明
IN/OUT	SInt, Int, DInt, USInt, UInt, UDInt	数値演算入力および出力

ENO のステ ータス	説明
1	エラーは発生していません。
0	結果の値が、選択したデータタイプの有効な範囲を超えています。
	SInt の場合の例: INC (127)の結果は-128 になり、データタイプ
	maximum の範囲を超えています。

絶対値命令



ABS 命令は、パラメータ IN の符号付き整数または実数の絶対値を取って、その結果をパラメータ OUT に格納するために使用します。 ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

IN および OUT パラメータは、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
IN	SInt, Int, DInt, Real, LReal	数値演算入力
OUT	SInt, Int, Dint, Real, LReal	数値演算出力

ENO のス テータス	説明
1	エラーは発生していません。
0	演算の結果の値が、選択したデータタイプの有効な範囲を超えていま す。
	SInt の場合の例: ABS (-128)の結果は+128 になり、データタイプ maximum の範囲を超えています。

MIN 命令と MAX 命令





次の場合には MIN (最小値)命令および MAX (最大値)命令を使用します。

- MIN 命令はパラメータ IN1 と IN2 の 2 つの値を比較し、最小(小さい方)の値をパラメータ OUT に代入します。
- MAX 命令はパラメータ IN1 と IN2 の 2 つの値を比較し、最大(大き い方)の値をパラメータ OUT に代入します。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

IN1、IN2、OUT の各パラメータは、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
IN1, IN2	SInt, Int, DInt, USInt, UInt, UDInt, Real, Constant	数值演算入力
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real	数値演算出力

ENO のステ ータス	説明
1	エラーは発生していません。
0	Real データタイプのみ
	一方または両方の入力がデータタイプ Real の数値ではありません (NaN)
	• 結果の OUT が+/- INF (無限大)です

上下限リミット命令

上下限リミット命令は、パラメータ IN の値がパラメータ MIN と MAX で指定した値の範囲に収まっているかどうかを調べるときに使用します。IN の値がこの範囲を超えている場合、OUT の値は MIN または MAX の値に固定されます。



- パラメータ IN の値が指定した範囲内に収まっている場合は、IN の値がパラメータ OUT に格納されます。
- パラメータ IN の値が指定した範囲を超えている場合は、OUT の値がパラメータ MIN の値(IN の値が MIN の値未満のとき)またはパラメータ MAX の値(IN の値が MAX の値を超えているとき)になります。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

注記

パラメータ MIN、IN、MAX、および OUT は、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
MIN, IN、およ び MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real, Constant	数値演算入力
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real	数値演算出力

ENO のステー タス	説明
1	エラーは発生していません。
0	Real: MIN、IN、MAX の 1 つ以上が NaN (非数)の場合は、NaN が返されます。
0	MIN が MAX よりも大きい場合は、値 IN が OUT に代入されます。

浮動小数点演算命令

浮動小数点演算命令は、Real または LReal データタイプの演算をプログラムするときに使用します。

- SQR: 2 乗(IN ² = OUT)
- SQRT: 平方根(√IN = OUT)
- LN: 自然対数(LN(IN) = OUT)
- EXP: 自然対数(e N = OUT)、ただし底 e = 2.71828182845904523536
- SIN: 正弦(sin(IN ラジアン) = OUT)
- COS: 余弦(cos(IN ラジアン) = OUT)
- TAN: 正接(tan(IN ラジアン) = OUT)
- ASIN: 逆正弦(arcsine(IN) = OUT ラジアン)、sin(OUT ラジアン) = IN
- ACOS: 逆余弦(arccos(IN) = OUT ラジアン)、cos(OUT ラジアン) = IN
- ATAN: 逆正接(arctan(IN) = OUT ラジアン)、tan(OUT ラジアン) = IN
- FRAC: 小数部(浮動小数点数 IN の小数部分 = OUT)
- EXPT: 一般指数(IN1 IN2 = OUT)



ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。 EXPT のパラメータ IN1 および OUT は常に Real です。 指数パラメータ IN2 のデータタイプを選択することができます。

	EXPT		
	Real ** ???		
-	EN	ENO	H
-	IN1	OUT	ŀ
-	IN2		

パラメータ	データタイプ	説明
IN, IN1	Real, LReal, Constant	入力
IN2	SInt, Int, Dint, USInt, UInt, UDInt, Real,	EXPT 指数入力
	LReal, Constant	
OUT	Real, LReal	出力

ENO のス テー タス	命令	条件	結果(OUT)
1	すべて	エラーは発生していません。	有効な結果
0	SQR	結果が Real/LReal の範囲を超えている	+INF
		IN が+/- NaN (非数)	+NaN
	SQRT	IN が負数	-NaN
		IN が+/- INF (無限大)または+/- NaN	+/- INF または+/- NaN
	LN	IN が 0.0、負数、-INF、または-NaN	-NaN
		IN が+INF または+NaN	+/- INF または+/- NaN
	EXP	結果が Real/LReal の範囲を超えている	+INF
		IN が+/- NaN	+/- NaN
	SIN、COS、 TAN	IN が+/- INF または+/- NaN	+/- INF または+/- NaN
	ASIN, ACOS	IN が有効な範囲の-1.0~+1.0 を超え ている	+NaN
		IN が+/- NaN	+/- NaN
	ATAN	IN が+/- NaN	+/- NaN
	FRAC	IN が+/- INF または+/- NaN	+NaN
	EXPT	IN1 が+INF で IN2 が-INF ではない	+INF
		IN1 が負数または-INF	IN2 が Real/LReal の ときは+NaN、そう でないときは -INF
		IN1 または IN2 が+/- NaN	+NaN
		IN1が 0.0で、IN2が Real/LReal (の み)	+NaN

6.1.6 ムーブ

ムーブ命令およびブロックのムーブ命令



ムーブ命令は、データ要素を新しいメモリアドレスにコピーして、データタイプの変換を行ったときに使用します。 ムーブ処理によって、元のデータに変更が加えられることはありません。

- MOVE: 指定したアドレスに格納されているデータ要素を、新しいアドレスにコピーします
- MOVE_BLK: 指定したデータ要素を新しいアドレスにコピーする、割り込み可能なムーブ命令
- UMOVE_BLK: 指定したデータ要素を新しいアドレスにコピーする割り込み不可能なムーブ命令

MOVE						
パラメータ	データタイプ	説明				
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, Array, Struct, DTL, Time	ソースアドレス				
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, Array, Struct, DTL, Time	宛先アドレス				

MOVE_BLK、UMOVE_BLK					
パラメータ	データタイプ	説明			
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte, Word, DWord	ソース先頭アドレス			
COUNT	UInt	コピーするデータ要素の数			
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte, Word, DWord	宛先先頭アドレス			

注記

データのコピー操作のルール

- ブールデータタイプのコピーには、SET_BF、RESET_BF、R、Sまたは出力コイル (LAD)を使用する
- 基本的な単一のデータタイプのコピーには、MOVE を使用する
- 基本的なデータタイプの配列のコピーには、MOVE_BLK または UMOVE_BLK を使用する
- 構造体のコピーには MOVE を使用する
- 文字列のコピーにはS CONV を使用する
- 文字列内の単一の文字のコピーには MOVE を使用する
- MOVE_BLK 命令および UMOVE_BLK 命令を使用して、配列または構造体を I、Q、Mメモリ領域にコピーすることはできません。

MOVE 命令は、IN パラメータで指定したソースアドレスから OUT パラメータで指定したターゲットアドレスに単一のデータ要素をコピーします。

MOVE_BLK 命令および UMOVE_BLK 命令には COUNT パラメータが存在します。 この COUNT パラメーを使用して、コピーするデータ要素の数を指定します。 コピーされる要素あたりのバイト数は、PLC タグテーブル内の IN および OUT パラメータタグの名前に割り付けられているデータタイプに依存します。

MOVE_BLK 命令と UMOVE_BLK 命令では割り込みの処理方法が異なります。

- MOVE_BLK の実行時には、割り込みイベントのキューへの格納および処理が行われます。 MOVE_BLK 命令は、割り込み OB サブプログラム内で移動先のアドレスを使用しないとき、または移動先のアドレスのデータの整合性が必要でないときに使用します。 MOVE_BLK での操作に割り込みが発生した場合、移動された最後のデータ要素は、移動先アドレスに移動されていて、一貫性が保持されています。 割り込み OB の実行が完了すると、MOVE_BLK の操作が再開されます。
- UMOVE_BLK命令の場合は、実行が完了するまでは、割り込みイベントのキューへの格納は行われますが、処理は行われません。 移動先のデータの一貫性を保持する 必要があるときには、割り込みOBの実行を開始する前に、移動操作が完了する UMOVE_BLK命令を使用します。 詳細については、「データの一貫性 (ページ 114)」のセクションを参照してください。

MOVE 命令の実行後、ENO は常に真です。

ENO のステ ータス	条件	結果
1	エラーは発生していません。	COUNT で指定したすべての要素が正常にコピーされました
0	移動元(IN)または移動先(OU)の範囲が使用可能なメモリ領域を超えています	範囲に収まる要素はコピーさ れています。 範囲に収まらな い要素はコピーされていませ ん。

フィル命令





FILL_BLK 命令および UFILL_BLK 命令は次のように使用します。

- FILL_BLK: この割り込み可能なフィル命令は、指定したデータ要素のコピーを使用して、あるアドレス範囲を充填します。
- UFILL_BLK: この割り込み不可能な充填命令は、指定したデータ要素のコピーを使用して、あるアドレス範囲を充填します。

パラメー タ	データタイプ	説明
IN	SInt, Int, DIntT, USInt, UInt, UDInt, Real, BYTE, Word, DWord	データのソースアドレス
COUNT	USInt, UInt	コピーするデータ要素の数
OUT	SInt, Int, DIntT, USInt, UInt, UDInt, Real, BYTE, Word, DWord	データ移動先アドレス

注記

データの充填操作のルール

- ブールデータタイプの充填には、SET_BF、RESET_BF、R、Sまたは出力コイル (LAD)を使用する
- 基本的な単一のデータタイプの充填には MOVE を使用する
- 基本的なデータタイプの配列の充填には FILL BLK または UFILL BLK を使用する
- 文字列内の単一の文字の充填には MOVE を使用する
- FILL_BLK 命令および UFILL_BLK 命令を使用して、I、Q、または M メモリ領域を配列で充填することはできません。

FILL_BLK 命令および UFILL_BLK 命令は、先頭アドレスをパラメータ OUT で指定した 移動先に、ソースのデータ要素 IN をコピーします。 コピープロセスは、コピー数が COUNT パラメータに等しくなるまで反復され、隣接アドレスの充填が行われます。

FILL_BLK 命令と UFILL_BLK 命令では割り込みの処理方法が異なります。

- FILL_BLK の実行時には、割り込みイベントのキューへの格納および処理が行われます。 FILL_BLK 命令は、割り込み OB サブプログラム内で充填するアドレスを使用しないとき、または充填するアドレスのデータの整合性が必要でないときに使用します。
- UFILL_BLK 命令の場合は、実行が完了するまでは、割り込みイベントのキューへの 格納は行われますが、処理は行われません。 移動先のデータの一貫性を保持する必 要があるときには、割り込み OB の実行を開始する前に移動操作が完了する UFILL BLK 命令を使用します。

ENO のス テータス	条件	結果
1	エラーは発生していません。	IN 要素は、COUNT で指定した回数、移動先に正常にコピーされました。
0	移動先(OUT)の範囲が、使用可能なメモリ領域を超えました	範囲に収まる要素はコピーされて います。 範囲に収まらない要素は コピーされていません。

6.1.6.1 スワップ命令



SWAP 命令は、2 バイトおよび 4 バイトのデータ要素のバイトの順序を反転するときに使用します。 各バイトのビットの順序に影響はありません。 SWAP 命令の実行後、ENO は常に真です。

パラメータ	データタイプ	説明
IN	Word, DWord	順序付きデータバイト IN
OUT	Word, DWord	順序を反転したデータバイト OUT

			メータ IN = MB0 実行前			例 パラメータ OUT = MB4 SWAP 実行後			
アドレス	MB0	MB 1				MB4	MB 5		
W#16#1234	12	34				34	12		
WORD	MSB	LSB				MS B	LSB		
アドレス	MB0	MB 1	MB 2	MB3		MB4	MB 5	MB 6	MB7
DW#16# 12345678	12	34	56	78		78	56	34	12
DWORD	MSB			LSB		MS B			LSB

6.1.7 変換

変換命令



CONVERT 命令は、データ要素のデータタイプの変換に使用します。 ボックス名の下をクリックし、ドロップダウンリストで IN および OUT のデータタイプを選択します。

ソースのデータタイプを選択すると、そのデータタイプから変換することのできるデータタイプがドロップダウンリストに表示されます。 変換前および変換後のデータタイプが BCD16 の場合は、変換後または変換前のデータタイプが Int に限定されます。 変換後または変換前のデータタイプが BCD32 の場合は、変換後または変換前のデータタイプが DInt に限定されます。

パラメー タ	データタイプ	説明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord, Real, LReal, Bcd16, Bcd32	IN 値
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord, Real, LReal, Bcd16, Bcd32	新しいデータタイプに変 換された IN 値

ENO のステ ータス	説明	結果の OUT
1	エラーは発生していません。	有効な結果
0	IN が+/- INF または+/- NaN	+/- INF または+/- NaN
0	結果が OUT データタイプの有効な範囲を超えている	OUT は、IN の下位バイトにセットされます。

丸め命令と切り捨て命令



ROUND は実数を整数に丸めます。 実数の小数点以下の部分は切り捨てられ、最も近い整数に丸められます(IEEE - 近似値への丸め)。 Real の数が 2 つの整数の中間値である場合は(10.5 など)、その Real の数は偶数に丸められます。 たとえば、ROUND (10.5) = 10 や ROUND (11.5) = 12 など。



TRUNC は実数を整数に変換します。 実数の小数点以下の部分は切り 捨てられゼロになります(IEEE - 0 への丸め)。

パラメータ	データタイプ	説明		
IN	Real, LReal	浮動小数点数入力		
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real,	丸められたか、または切り捨		
	LReal	てられた出力		

ENO のス テータス	説明	結果の OUT
1	エラーは発生していません。	有効な結果
0	IN が+/- INF または+/- NaN	+/- INF または+/- NaN

切り上げ命令と切り下げ命令



CEIL は、実数をその実数以上の最も小さい整数に変換します(IEEE - 正の無限大への丸め)。



FLOOR は、実数をその実数以下の最も大きい整数に変換します(IEEE - 負の無限大への丸め)。

パラメータ	データタイプ	説明
IN	Real, LReal	浮動小数点数入力
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	変換された出力

ENO のス テータス	説明	結果の OUT
1	エラーは発生していません。	有効な結果
0	IN が+/- INF または+/- NaN	+/- INF または+/- NaN

6.1.7.1 スケーリング命令と正規化命令

スケーリング命令と正規化命令



SCALE_X は、スケーリングされた実数パラメータ VALUE ($0.0 \le VALUE \le 1.0$)を、MIN および MAX パラメータで指定したデータタイプおよび値の範囲にスケーリングします。

OUT = VALUE (MAX - MIN) + MIN

 $SCALE_X$ の場合、パラメータ MIN、IN、MAX、および OUT は、同じデータタイプでなければなりません。



NORM_X は、パラメータ VALUE を MIN および MAX パラメータで指定 した値の範囲に正規化します。

OUT = (VALUE - MIN) / (MAX - MIN)、ただし $0.0 \le 0$ OUT ≤ 1.0 NORM_X の場合、パラメータ MIN、VALUE、および MAX は、同じデータタイプでなければなりません。

パラメータ	データタイプ	説明
MIN	SInt, Int, DInt, USInt, UInt, UDInt, Real	範囲の入力最小値
VALUE	SCALE_X: REAL	スケーリングまたは正規化す
	NORM_X: SInt, Int, DInt, USInt, UInt,	る入力値
	UDInt, Real	
MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real	範囲の入力最大値
OUT	SCALE_X: SInt, Int, DInt, USInt, UInt,	スケーリングまたは正規化さ
	UDInt, Real	れた出力値
	NORM_X: REAL	

注記

SCALE_X のパラメータ VALUE は($0.0 \le VALUE \le 1.0$)の範囲でなければなりません。

パラメータ VALUE が 0.0 未満または 1.0 を超過している場合

- この線形スケーリング演算は、OUT のデータタイプの範囲内に収まる MIN 値未満または MAX 値を超える OUT 値を生成することができます。 この場合、SCALE_X の実行によって ENO = TRUE にセットされます。
- OUT のデータタイプの範囲内に収まらないスケーリングされた数を生成することができます。 この場合、パラメータ OUT の値は、OUT のデータタイプへの最終変換の前に、スケーリングされた実数の下位部分に等しい中間値にセットされます。 この場合、SCALE_X の実行によって ENO = FALSE にセットされます。

NORM_X のパラメータ VALUE は(MIN <= VALUE <= MAX)の範囲でなければなりません。

パラメータ VALUE が MIN 未満または MAX を超過している場合、この線形スケーリング演算は、0.0 未満または 1.0 を超える正規化された OUT 値を生成することができます。 この場合、NORM_X の実行によって ENO = TRUE にセットされます。

ENO のス テータス	条件	結果の OUT
1	エラーは発生していません。	有効な結果
0	結果が OUT データタイプの 有効な範囲を超えている	中間結果: OUT データタイプへの最終変 換の前の実数の下位部分
0	パラメータ MAX <= MIN	SCALE_X: OUT のサイズに充填される Real の数の下位部分
		NORM_X: ダブルワードサイズを満たす ために拡大された VALUE データタイプ の VALUE
0	パラメータ VALUE = +/- INF または+/- NaN	VALUE が OUT に書き込まれる

6.1.8 プログラムコントロール

ジャンプ命令とラベル命令

実行シーケンスの条件制御にはプログラム制御命令を使用します。



パラメータ	データタイプ	説明
Label_name	ラベル識別子	ジャンプ命令用の識別子および対応する移動先
		プログラムラベル

LABEL 命令に直接入力することによって、ラベル名を作成します。 パラメータヘルパアイコンを使用して、JMP および JMPN ラベル名フィールドに使用可能なラベル名を選択することができます。 また、JMP および JMPN 命令に、直接ラベル名を入力することもできます。

Return_Value (RET)実行制御命令

LAD FBD "Return_Value" "Return_Value" —(RET)— RET

現在のブロックの実行を終了するときに RET 命令を使用します。

パラメータ	データタ イプ	説明
Return_Value	Bool	Return_value パラメータ(RET 命令)は、呼び出し元のブロックの ENO 出力に代入されます。

オプションとして使用する RET 命令は、現在のブロックの実行を終了するときに使用します。 RET コイル(LAD)への信号状態が ON の場合、または RET ボックス入力が真の場合(FBD)、現在のブロックのプログラム実行がそのポイントで終了し、RET 命令以降の命令は実行されません。 現在のブロックが OB の場合は、「Return_Value」パラメータは無視されます。 現在のブロックが FC または FB の場合、「Return_Value」パラメータの値は、呼び出されたブロックの ENO 値として、呼び出し元のルーチンに返されます。.

ブロックの最後の命令として RET 命令を使用する必要はありません。これは自動的に 行われます。 単一のブロックに複数の RET 命令を配置することができます。

FC プログラムブロック内で RET 命令を使用する手順の例を下記に示します。

- 1. 新しいプロジェクトを作成し、FCを1つ追加します。
- 2. 追加した FC を編集します。
 - 命令ツリーから命令を追加します。
 - RET 命令を追加します。「Return_Value」パラメータとして、
 TRUE、FALSE、または必要な戻り値が格納されているメモリロケーションを指定します。
 - 必要な命令を追加します。
- **3**. この FC を MAIN [OB1]から呼び出します。

この FC の実行を開始するには、MAIN プログラムブロック内のこの FC ボックスの EN 入力が真でなければなりません。

RET 命令への信号状態が ON である FC の実行後に、この FC 内の RET 命令で指定された値が、MAIN プログラムブロック内の FC ボックスの ENO 出力に格納されています。

6.1.9 論理演算

AND、OR、XOR 命令



AND: BYTE、WORD、および DWORD データタイプの論理 AND OR: BYTE、WORD、および DWORD データタイプの論理 OR XOR: BYTE、WORD、および DWORD データタイプの排他的論理 OR ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

パラメータ	データタイプ	説明
IN1、IN2	Byte, Word, DWord	論理入力
OUT	Byte, Word, DWord	論理出力

データタイプの選択によって、IN1、IN2、OUTが同じデータタイプに設定されます。 IN1 と IN2 の対応するビット値が組み合わされたバイナリの論理結果がパラメータ OUT に生成されます。 これらの命令の実行後、ENO は常に TRUE です。

反転命令



INV 命令はパラメータ IN の 2 進数 1 の補数を取るために使用します。 1 の補数は IN パラメータの各ビット値を反転することによって生成されます(それぞれの値の 0 を 1 に、1 を 0 に変換)。 この命令の実行後、ENO は常に TRUE です。

パラメータ	データタイプ	説明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	反転させるデータ要素
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	反転された出力

エンコード命令とデコード命令





ENCO は、ビットパターンをエンコードして 2 進数に変換します。

DECO は、2 進数をデコードしてビットパターンに変換します。

ボックス名の下をクリックし、ドロップダウンメニューでデータタイプを選択します。

パラメータ	データタイプ	説明
IN	ENCO: Byte, Word, DWord DECO: UInt	ENCO: エンコードするビットパターン
		DECO: デコードする値
OUT	ENCO: Int	ENCO: エンコードした値
	DECO: Byte, Word, DWord	DECO: デコードしたビットパタ
		ーン

ENCO 命令は、パラメータ IN の下位ビットのビット位置に基づいてパラメータ IN を 2 進数に変換し、その結果をパラメータ OUT に返します。 パラメータ IN が 0000 0001 または 0000 0000 の場合は値 0 が OUT に返されます。 パラメータ IN の値が 0000 0000 の場合は ENO が FALSE にセットされます。

DECO 命令は、パラメータ IN から取得した 2 進数を、パラメータ OUT 内の対応する ビット位置を 1 にセットすることによってデコードします(その他のビットはすべて 0 にセットされます)。 DECO 命令の実行後、ENO は常に TRUE です。

DECO のパラメータ OUT のデータタイプが Byte、Word、または DWord によって、パラメータ IN の有効な範囲が決定されます。パラメータ IN の値が有効な範囲を超えた場合は、剰余算演算が実行され、下に示す最下位ビットが抽出されます。

DECO のパラメータ IN の範囲

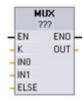
- **3** ビット(値 **0~7**)の **IN** が、データタイプがバイトの **OUT** の **1** ビットの位置に使用 されます。
- **4** ビット(値 **0~15**)の IN が、データタイプがワードの **OUT** の **1** ビットの位置に使用されます。
- 5 ビット(値 0~31)の IN が、データタイプがダブルワードの OUT の 1 ビットの位置に使用されます。

DECO の IN	の値	DECO の OUT の値(単一のビット位置のデコード)
		Byte の OUT (8 ビット)
IN の最小 値	0	0000001
IN の最大 値	7	10000000
		Word の OUT (16 ビット)
IN の最小 値	0	0000000000001
IN の最大 値	15	10000000000000
		DWord の OUT (32 ビット)
IN の最小 値	0	000000000000000000000000000000000000000
IN の最大 値	31	100000000000000000000000000000000000000

ENO のステータス	条件	結果(OUT)
1	エラーは発生していません。	有効なビット数値
0	IN がゼロ	OUT がゼロにセットされる

選択(SEL)命令と多重化(MUX)命令





- SEL は、パラメータ G の値に基づいて、2 つの入力値の 1 つをパラメータ OUT に代入します。
- MUX は、パラメータ K の値に基づいて、多数の入力値の 1 つをパラメータ OUT に代入します。 パラメータ K の値が有効な範囲を超えた場合は、パラメータ ELSE の値がパラメータ OUT に代入されます。

SEL	データタイプ	説明
G	BOOL	セレクタスイッチ
		INO の場合は
		FALSE
		• IN1 の場合は TRUE
INO、IN1	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte,	入力
	Word, DWord, Time, Char	
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte,	出力
	Word, DWord, Time, Char	

MUX	データタイプ	説明
К	UInt	セレクタ値 • IN0 の場合は 0 • IN1 の場合は 1 •
IN0、 IN1、	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte, Word, DWord, Time, Char	入力
Else	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte, Word, DWord, Time, Char	入力代替値(オプショ ン)
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, Byte, Word, DWord, Time, Char	出力

入力変数および出力変数は、同じデータタイプでなければなりません。

- **SEL** 命令は常に **2** つの **IN** 値の **1** つを選択します。
- MUX 命令には、初めてプログラムエディタに配置したときには IN パラメータが 2 つ存在しますが、拡大して IN パラメータをさらに追加することができます。

MUX 命令の入力パラメータを追加または削除するには、次の方法を使用します。

- 入力を追加するには、既存の IN パラメータの入力スタブを右クリックし、[Insert input]を選択します。
- 入力を削除するには、既存の IN パラメータの入力スタブを右クリックし(元の 2 つの入力以外に追加した入力が存在する場合)、[Delete]を選択します。

条件コード: SEL 命令の実行後、ENO は常に TRUE です。

ENO のステータス (MUX)	MUX の条件	MUX の結果の OUT
1	エラーは発生していません。	選択した IN 値が OUT に代入される
0	KがINパラメータの数以上	ELSE を指定していない場合: OUT は変化しない
		ELSE を指定した場合: ELSE の値が OUT に代入される

6.1.10 シフトおよび回転

シフト命令



パラメータ IN のビットパターンをシフトするときにシフト命令を使用します。結果はパラメータ OUT に代入されます。 パラメータ N は、シフトするビット位置の数を指定します。

- SHR: ビットパターンを右にシフトします。
- SHL: ビットパターンを左にシフトします。

パラメータ	データタイプ	説明
IN	Byte, Word, DWord	シフトさせるビットパターン
N	UInt	シフトさせるビット位置の数
OUT	Byte, Word, DWord	シフト操作後のビットパターン

- N=0 の場合はシフトは行われず、IN の値が OUT に代入されます。
- シフト操作によって空になったビット位置はゼロで埋められます。
- シフトさせる位置の数(N)が目標値のビット数(Byte の場合は 8、Word の場合は 16、 DWord の場合は 32)を超えている場合は、元のすべてのビット値がゼロに置換されます(OUT にはゼロが代入されます)。
- シフト演算の場合 ENO は常に TRUE です。

Wor	Word サイズのデータの SHL の例: 左側へのゼロのシフト			
IN 1110 0010 1010 1101 最初のシフトの前の OUT の値		最初のシフトの前の OUT の 値	1110 0010 1010 1101	
		最初の左シフトの後	1100 0101 0101 1010	
		2番目の左シフトの後	1000 1010 1011 0100	
		3番目の左シフトの後	0001 0101 0110 1000	

ローテーション命令



パラメータ IN のビットパターンを回転させるにはローテーション命令を 使用します。結果はパラメータ OUT に代入されます。 パラメータ N は、回転させるビット位置の数を定義します。

- ROR: ビットパターンを右に回転させます
- ROL: ビットパターンを左に回転させます

パラメータ	データタイプ	説明
IN	Byte, Word, DWord	回転させるビットパターン
N	UInt	シフトさせる数
OUT	Byte, Word, DWord	回転操作後のビットパターン

- N=0 の場合は回転は行われず、IN の値が OUT に代入されます。
- 回転のためにターゲット値の片側からはみ出したビットはターゲット値の反対側に 回り込むため、元のビット値は失われません。
- 回転させる数がターゲット値のビット数(Byte の場合は 8、Word の場合は 16、 DWord の場合は 32)を超えている場合でも、回転が実行されます。
- ローテーション命令の実行後、ENO は常に TRUE です。

WOF	WORD サイズのデータの ROR の例: 右側から左側へビットを回転させる			
IN	0100 0000 0000 0001	最初の回転の前の OUT の値	0100 0000 0000 0001	
		最初の右回転の後	1010 0000 0000 0000	
		2番目の右回転の後	0101 0000 0000 0000	

6.2 拡張命令

6.2.1 拡張命令用の共通のエラーパラメータ

拡張命令の記述には、それぞれのプログラム命令で発生する可能性のある実行時エラーを記述します。 これらのエラー以外に、下の表に示す共通エラーも使用することができます。 プログラムブロックが実行されたときに共通エラーの 1 つが発生すると、そのプログラムブロック内で GetError または GetErrorID 命令を使用して、エラーに対する応答をプログラムしていない限り、CPU は STOP モードに移行します。

条件コードの値(W#16#)	説明
8022	入力に対して領域が小さすぎる
8023	出力に対して領域が小さすぎる
8024	不正な入力領域
8025	不正な出力領域
8028	不正な入力ビット割り付け
8029	不正な出力ビット割り付け
8030	出力領域が読み取り専用 DB
803A	DB が存在しない

6.2 拡張命令

6.2.2 クロックおよびカレンダ命令

日付と時刻の命令

日付と時刻の命令は、カレンダおよび時間の計算に使用します。

- T_CONV は時刻値のデータタイプを変換します: (Time を DInt に)または(DInt を Time に)
- T_ADD は Time と DTL の値の加算を行います: (Time + Time = Time)または(DTL + Time = DTL)
- T_SUB は Time と DTL の値の減算を行います: (Time Time = Time)または(DTL Time = DTL)
- T_DIFF は、2 つの DTL 値の差を求めて Time 値に変換します: DTL DTL = Time

データタイプ	サイズ(ビッ ト)	有効範囲
Time	32	T#-24d_20h_31m_23s_648ms~ T#24d_20h_31m_23s_647ms
	(格納値)	-2,147,483,648 ms~+2,147,483,647 ms
DTL データ構造体		
Year: UInt	16	1970~2554
Month: USInt	8	1~12
Day: USInt	8	1~31
Weekday: USInt	8	1(日曜日)~7(土曜日)
Hour: USInt	8	0~23
Minute: USInt	8	0~59
Second: USInt	8	0~59
Nanoseconds: UDInt	32	0~999,999,999



T_CONV (Time Convert)は、Time データタイプを DInt データタイプ に、またはその逆に DInt データタイプを Time データタイプに変換します。

パラメータ	パラメータ タイプ	データタイプ	説明
IN	IN	DInt、Time	入力された Time 値または DInt 値
OUT	OUT	DInt、Time	変換された Time 値または DInt 値

命令名の下をクリックし、ドロップダウンリストで、IN および OUT のデータタイプを選択します。



T_ADD (Time Add)は、IN1 値(DTL または Time データタイプ)と IN2 の Time 値を加算します。 パラメータ OUT は DTL または Time データタイプの結果の値です。

2つのデータタイプの演算が可能です

- Time + Time = Time
- DTL + Time = DTL

パラメータ	パラメータ タイプ	データタイプ	説明
IN1	IN	DTL、Time	DTL 値または Time 値
IN2	IN	Time	加算する Time 値
OUT	OUT	DTL、Time	DTL または Time データタイプの和

命令名の下をクリックし、ドロップダウンリストで IN1 のデータタイプを選択します。 IN1 のデータタイプを選択すると、パラメータ OUT のデータタイプもセットされます。



T_SUB (Time Subtract)は、IN1 (DTL 値または Time 値)から IN2 の Time 値を減算します。 パラメータ OUT は演算結果の差の DTL または Time データタイプの値です。

2つのデータタイプの演算が可能です

- Time Time = Time
- DTL Time = DTL

パラメータ	パラメータ タイプ	データタイプ	説明
IN1	IN	DTL、Time	DTL 値または Time 値
IN2	IN	Time	減算する Time 値
OUT	OUT	DTL、Time	DTL または Time データタイプの差

命令名の下をクリックし、ドロップダウンリストで IN1 のデータタイプを選択します。 IN1 のデータタイプを選択すると、パラメータ OUT のデータタイプもセットされます。



T_DIFF (Time Difference)は、IN1 の DTL 値から IN2 の DTL 値を減算します。 パラメータ OUT は演算結果の差の Time データタイプの値です。

• DTL - DTL = Time

パラメータ	パラメータ タイプ	データタイプ	説明
IN1	IN	DTL	DTL 値
IN2	IN	DTL	減算する DTL 値
OUT	OUT	Time	差の Time 値

条件コード: ENO = 1 はエラーが発生しなかったことを示します。 ENO = 0 およびパラメータ OUT = 0 はエラーです。

- 無効な DTL 値
- 無効な Time 値

クロック命令

PLC システムクロックの設定および読み取りには、クロック命令を使用します。 日付 と時刻の値にはデータタイプ DTL が使用されます。

DTL 構造体	サイズ	有効範囲
Year: UInt	16 ビット	1970~2554
Month: USInt	8 ビット	1~12
Day: USInt	8 ビット	1~31
Weekday: USInt	8 ビット	1(日曜日)~7(土曜日)
Hour: USInt	8 ビット	0~23
Minute: USInt	8 ビット	0~59
Second: USInt	8 ビット	0~59
Nanoseconds: UDInt	32 ビット	0~999,999,999



WR_SYS_T (システム時刻を書き込む)は、PLC の時刻をパラメータ IN の DTL 値にセットします。この値には、ローカル時間帯もサマータイムオフセットも含まれません。

パラメータ	パラメータ タイプ	データタ イプ	説明
IN	IN	DTL	PLC システムクロックにセットする時刻
RET_VAL	OUT	Int	実行条件コード



RD_SYS_T (システム時刻を読み取る)は、現在の時刻を PLC から読み取ります。 この値には、ローカル時間帯もサマータイムオフセットも含まれません。

パラメータ	パラメータ タイプ	データタ イプ	説明
RET_VAL	OUT	Int	実行条件コード
OUT	OUT	DTL	現在の PLC システム時刻



RD_LOC_T (ローカル時刻を読み取る)は、PLC の現在のローカル時刻をDTL データタイプの値として取得します。

パラメータ	パラメータ タイプ	データタ イプ	説明
RET_VAL	OUT	Int	実行条件コード
OUT	OUT	DTL	ローカル時刻

- ローカル時刻は、[CPU Clock]デバイス構成で設定したタイムゾーンとサマータイム オフセットを使用して計算されます。
- 時間帯の構成は、協定世界時(UTC)のシステム時刻に対するオフセットの指定です。
- サマータイムの設定は、サマータイムが開始される月、週、日、時刻の指定です。
- 標準時刻の構成は、標準時刻が開始される月、週、日、時刻の指定です。
- 時間帯オフセットは常にシステム時刻の値に適用されます。 サマータイムオフセットは、サマータイムが実施されたときにのみ適用されます。

条件コード: ENO = 1 はエラーが発生しなかったことを示します。 **ENO = 0** は実行エラーが発生したことを意味し、条件コードが **RET_VAL** 出力に格納されます。

RET_VAL (W#16#)	説明
0000	エラーは発生していません。
8080	ローカル時刻が使用できない
8081	年の値不正
8082	月の値不正
8083	日の値不正
8084	時の値不正
8085	分の値不正
8086	秒の値不正
8087	ナノ秒の値不正
80B0	リアルタイムクロック異常

6.2.3 文字列命令と文字命令

6.2.3.1 文字列データの概要

文字列データタイプ

文字列データは、2 バイトのヘッダと、それに続く最大 254 文字バイトの ASCII キャラクタコードで構成されます。 文字列ヘッダには、2 つのデータ長が含まれています。最初のバイトは、文字列を初期化したときに大括弧で囲んで指定する最大長で、デフォルトでは 254 です。 2 番目のヘッダバイトは現在のデータ長で、その文字列内の有効な文字の数です。 現在のデータ長は最大データ長以下でなければなりません。 文字列形式で格納される占有バイト数は、最大長より 2 バイト多くなります。

文字列データを初期化する

文字列命令を実行する前に、文字列入力および出力をメモリ内の有効な文字列として初期化しておく必要があります。

有効な文字列データ

文字列の長さは 0 よりも大きく、255 よりも小さい値です。現在のデータ長は最大長以下でなければなりません。

文字列を I および Q メモリ領域に割り付けることはできません。

詳細については、以下を参照してください。「Stringデータタイプのフォーマット」 (69ページ) (ページ 76)

6.2.3.2 文字列変換命令

文字列を値へ、値を文字列への変換

これらの命令を使用して、数字の文字列を数値に、数値を数字の文字列に変換することができます。

- S CONV は、「文字列を数値に」または「数値を文字列に」変換します。
- STRG_VAL は、フォーマットオプションを使用して、数字の文字列を数値に変換します。
- STRG_VAL は、フォーマットオプションを使用して、数値を数字の文字列に変換します。



S_CONV (文字列変換)は、文字列を対応する値に、値を対応する文字列に変換します。 **S_CONV** 命令には出力フォーマットオプションはありません。 **S_CONV** 命令の方が単純ですが、**STRG_VAL** および**VAL_STRG** のように柔軟な使い方はできません。

ドロップダウンリストでパラメータのデータタイプを選択します。

S_CONV (文字列から値への変換)

パラメー タ	パラメータタイ プ	データタイプ	説明
IN	IN	String	入力文字列
OUT	OUT	String, SInt, Int, DInt, USInt, UInt, UDInt, Real	出力数値

文字列パラメータ IN の変換は最初の文字から始まり、その文字列の終りまで、または、「0~9」、「+」、「-」、「.」ではない最初の文字が見つかるまで続行されます。 結果の値は、OUT で指定したロケーションに格納されます。 出力パラメータの値が OUT データタイプの範囲に収まらない場合は、パラメータ OUT は 0 にセットされ、ENO が FALSE にセットされます。 そうでない場合は、パラメータ OUT に有効な結果が格納され、ENO が TRUE にセットされます。

入力文字列のフォーマットのルール

- IN 文字列に小数点を使用する場合は、ピリオド「.」文字を使用する必要があります。
- 小数点から左側の3桁の区切りのコンマ「,」文字は許されますが、無視されます。
- 先頭の空白は無視されます。
- サポートされているのは、固定小数点表記のみです。 文字「e」および「E」は、指数表記として認識されません。

S_CONV (値から文字列への変換)

パラメー	パラメータタイ プ	データタイプ	説明
タ	プ		
IN	IN	String, SInt, Int, DInt, USInt,	入力数値
		Ulnt, UDInt, Real	
OUT	OUT	String	出力文字列

整数、符号なし整数、浮動小数点数の IN を対応する文字列に変換して、OUT に格納します。 変換を実行する前に、パラメータ OUT が有効な文字列を参照している必要があります。 有効な文字列は、最大長(最初のバイト)、現在のデータ長(2 番目のバイト)、現在の文字列の文字(以降のバイト)で構成されます。 OUT 文字列内の文字が最初の文字から変換された文字列の文字に置換され、OUT 文字列の現在のデータ長が調整されます。 OUT 文字列の最大長は変更されません。

置換される文字数は、パラメータ IN のデータタイプと元の数値に依存します。 置換される文字数が、パラメータ OUT 文字列の最大長以内でなければなりません。 OUT 文字列の最大長(最初のバイト)が、変換後の最大文字数以上でなければなりません。 サポートされている各データタイプに必要な文字列の最大長を下の表に示します。

IN データ タイプ	変換後に OUT 文字列に格納 される最大文字数	例	最大長および現在のデータ長 が含まれた文字列の合計の長 さ
USInt	3	255	5
SInt	4	-128	6
UInt	5	65535	7
Int	6	-32768	8
UDInt	10	429496729	12
		5	
DInt	11	-	13
		214748364	
		8	

出力文字列のフォーマットのルール

- パラメータ OUT に書き込まれる値の先頭に「+」記号は使用されません。
- 固定小数点表記が使用されます(指数表記は使用されません)。
- パラメータ IN が Real データタイプのときは、小数点の表記にピリオド文字「.」が 使用されます。

STRG_VAL 命令



STRG_VAL (文字列から値へ)は、数字の文字列を対応する整数または浮動小数点表現に変換します。 変換は文字列 IN のキャラクタオフセット P の位置から始まり、その文字列の終りまで、または、「+」、「-」、「.」、「,」、「e」、「E」、「0~9」ではない最初の文字が見つかるまで続行されます。 結果はパラメータ OUT で指定したロケーションに格納されます。

パラメータ P も、元の文字列の変換が終了した位置のオフセットカウントとして返されます。 変換を実行する前に、文字列データが、メモリ内の有効な文字列として初期化されている必要があります。

パラメー タ	パラメータタ イプ	データタイプ	説明
IN	IN	String	変換する ASCII 文字列
FORMAT	IN	Word	出力フォーマットオプション
P	IN_OUT	UInt	IN: 変換する最初の文字のインデックス(最初の文字 = 1)OUT: 変換プロセスが終了した後の次の文字のインデックス
OUT	OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real	変換後の数値

STRG_VAL の FORMAT パラメータ

STRG_VALのFORMATパラメータは、次のように定義されます。 未使用のビット位置はゼロにセットされている必要があります。

ト 16 0	_	0	_	0	0	ト 8 0	ኑ 7	0	0	0	_	_	_	ኑ 0
ッ						ッ	ッ							ッ
Ľ						ピ	ビ							Ŀ

f = 表記フォーマット 1= 指数表記

0 = 固定小数点表記

r = 小数点のフォーマット 1 = "," (コンマ文字)

0 = "." (ピリオド文字)

FORMAT (W#16#)	表記フォーマット	小数点の表現
0000 (デフォルト)	固定小数点	п п
0001		н н
0002	指数	н н
0003		и и ,
0004~FFFF	値が不正です。	

STRG VAL の変換のルール

- 小数点にピリオド文字「.」を使用した場合は、小数点の左側に存在するコンマ 「,」は3桁の区切り文字として解釈されます。 コンマ文字を使用することはでき ますが、無視されます。
- 小数点にコンマ文字「,」を使用した場合は、小数点の左側に存在するピリオド「.」は3桁の区切り文字として解釈されます。 ピリオド文字を使用することはできますが、無視されます。
- 先頭の空白は無視されます。

VAL_STRG 命令



VAL_STRG (値から文字列へ)は、整数、符号なし整数、または浮動小数点数を対応する文字列表現に変換します。 パラメータ IN で表された値が、パラメータ OUT によって参照される文字列に変換されます。 変換を実行する前に、パラメータ OUT が有効な文字列を参照している必要があります。

OUT 文字列内のキャラクタオフセットカウント P の位置からパラメータ SIZE によって指定される文字数が、変換後の文字列に置換されます。 文字位置 P から数えた文字数 SIZE が OUT 文字列の長さ以下でなければなりません。この命令は、テキスト文字列に数字を埋め込むのに便利です。 たとえば、数字「120」を文字列「Pump pressure = 120 psi」に埋め込むことができます。

パラメー タ	パラメータ タイプ	データタイプ	説明
IN	IN	SInt, Int, DInt, USInt, UInt, UDInt, Real	文字列に変換する値
サイズ	IN	USInt	OUT 文字列に書き込む文字数
PREC	IN	USInt	精度(小数点以下の部分のサイズ)。 小数点は含まず。
フォーマット	IN	WORD	出力フォーマットオプション
Р	IN_OUT	UInt	IN: OUT 文字列内の置換する最初の 文字のインデックス(先頭の文字 = 1)OUT: 置換後の OUT 文字列内の次の 文字のインデックス
OUT	OUT	文字列	変換後の文字列

パラメータ PREC は、精度(文字列の小数点以下の部分の桁数)を指定します。 パラメータ IN の値が整数の場合、PREC は小数点の位置を指定します。 たとえば、データ値が 123 で PREC = 1 の場合、結果は「12.3」になります。 サポートされている REAL データタイプの最大精度は 7 桁です。

パラメータ P が OUT 文字列の現在の長さよりも大きいときは、位置 P まで空白が追加され、結果が文字列の終りに追加されます。 OUT 文字列の最大長に達すると変換が終了します。

VAL_STRG の FORMAT パラメータ

VAL_STRG の FORMAT パラメータは、次のように定義されます。 未使用のビット位置はゼロにセットされている必要があります。

ト 16 0		0	0	0	ト 8 0	ኑ 7	0	0	_		_	۲ 0
ッ					ツ	ツ						ッ
Ľ					ビ	ビ						F.

s = 符号文字 **1**= 符号の「+」と「-」を使用する

0 = 符号「-」のみを使用する

f = 表記フォーマット **1** = 指数表記

0 = 固定小数点表記

r = 小数点のフォーマット 1 = "," (コンマ文字)

0 = "." (ピリオド文字)

FORMAT (WORD)	符号	表記フォーマット	小数点の表現
W#16#0000	「-」のみ	固定小数点	"."
W#16#0001			II II
W#16#0002		指数	" "
W#16#0003			п п ,
W#16#0004	「+」および「-」	固定小数点	" "
W#16#0005			п п ,
W#16#0006		指数	""
W#16#0007			п п ,
W#16#0008~ W#16#FFFF	値が不正です。		

パラメータ OUT 文字列のフォーマットのルール

- 変換後の文字列が指定したサイズよりも小さいときは、文字列の左側に空白が追加されます。
- FORMAT パラメータの符号ビットが FALSE のとき、符号なしおよび符号付き整数 データタイプの値は、先頭の「+」の符号を使用せずに出力バッファに書き込まれます。 必要に応じて「-」の符号は使用されます。
 - <先頭の空白><先頭の0を削除した数字>'.'<PREC 桁数の数字>
- 符号ビットが TRUE のときは、符号なしおよび符号付き整数データタイプの値は常に先頭の符号付きで出力バッファに書き込まれます。 先頭の空白><符号><先頭の 0 を削除した数字>'.'<PREC 桁数の数字>
- FORMAT が指数表記にセットされているときは、REAL データタイプの値は次のフォーマットで書き込まれます。
 <先頭の空白><符号><数字>'.' <PREC 桁数の数字>'E' <符号><先頭の 0 を削除した数字>
- FORMAT が固定小数点表記にセットされているときに、整数、符号なし整数、実数 データタイプの値は次のフォーマットで書き込まれます。 <先頭の空白><符号><先頭の 0 を削除した数字>'.'<PREC 桁数の数字>
- 小数点の左側の先頭の 0 は(小数点のすぐ左側のものを除き)、削除されます。
- 小数点の右側の値は、PREC パラメータで指定した小数点以下の桁数に収まるよう に丸められます。
- 出力文字列のサイズは、小数点以下の桁数よりも3バイト以上大きいサイズでなければなりません。
- 値は出力文字列に右揃えにされます。

ENO によって報告される条件

変換操作でエラーが発生したときは、下記の結果が返されます。

- **ENO** が **0** にセットされます。
- OUTが 0にセットされるか、文字列から値への変換例のようにセットされます。
- OUT が変化しないか、OUT が文字列のときの例のようにセットされます。

ENO のス テータス	説明
1	エラーは発生していません。
0	不正または無効なパラメータ(存在しない DB へのアクセスなど)
0	最大長が 0 または 255 の不正な文字列
0	現在の長さが最大長よりも長い不正な文字列
0	変換後の数値が指定した OUT データタイプに対して大き過ぎる
0	OUT パラメータの文字列の最大サイズは、文字位置パラメータ P からパラメータ SIZE で指定した文字数を収容できる大きさでなければなりません
0	P=0、または現在の文字列の長さを超えている不正な P 値
0	パラメータ SIZE はパラメータ PREC よりも大きい値でなければなりません

S_CONVによる文字列から値への変換の例

IN 文字列	OUT データタイプ	OUT の値	ENO
"123"	Int/DInt	123	TRUE
"-00456"	Int/DInt	-456	TRUE
"123.45"	Int/DInt	123	TRUE
"+2345"	Int/DInt	2345	TRUE
"00123AB"	Int/DInt	123	TRUE
"123"	REAL	123.0	TRUE
"123.45"	REAL	123.45	TRUE
"1.23e-4"	REAL	1.23	TRUE
"1.23e-4"	REAL	1.23	TRUE
"12,345.67"	REAL	12345.67	TRUE
"3.4e39"	REAL	3.4	TRUE
"-3.4e39"	REAL	-3.4	TRUE
"1.17549e-38"	REAL	1.17549	TRUE
"12345"	SInt	0	FALSE
"A123"	N/A	0	FALSE
	N/A	0	FALSE
"++123"	N/A	0	FALSE
"+-123"	N/A	0	FALSE

S_CONV による値から文字列への変換の例

データタイプ	IN 値	OUT 文字列	ENO
UInt	123	"123"	TRUE
UInt	0	"0"	TRUE
UDInt	12345678	"12345678"	TRUE
REAL	-INF	"INF"	FALSE
REAL	+INF	"INF"	FALSE
REAL	NaN	"NaN"	FALSE

STRG_VAL 変換の例

IN 文字列	FORMAT (W#16#)	OUT データタイ プ	OUT の値	ENO
"123"	0000	Int/DInt	123	TRUE
"-00456"	0000	Int/DInt	-456	TRUE
"123.45"	0000	Int/DInt	123	TRUE
"+2345"	0000	Int/DInt	2345	TRUE
"00123AB"	0000	Int/DInt	123	TRUE
"123"	0000	REAL	123.0	TRUE
"-00456"	0001	REAL	-456.0	TRUE
"+00456"	0001	REAL	456.0	TRUE
"123.45"	0000	REAL	123.45	TRUE
"123.45"	0001	REAL	12345.0	TRUE
"123,45"	0000	REAL	12345.0	TRUE
"123,45"	0001	REAL	123.45	TRUE
".00123AB"	0001	REAL	123.0	TRUE
"1.23e-4"	0000	REAL	1.23	TRUE
"1.23E-4"	0000	REAL	1.23	TRUE
"1.23E-4"	0002	REAL	1.23E-4	TRUE
"12,345.67"	0000	REAL	12345.67	TRUE

IN 文字列	FORMAT (W#16#)	OUT データタイ プ	OUT の値	ENO
"12,345.67"	0001	REAL	12.345	TRUE
"3.4e39"	0002	REAL	+INF	TRUE
"-3.4e39"	0002	REAL	-INF	TRUE
"1.1754943e-38" (およびそれ以下)	0002	REAL	0.0	TRUE
"12345"	N/A	SInt	0	FALSE
"A123"	N/A	N/A	0	FALSE
""	N/A	N/A	0	FALSE
"++123"	N/A	N/A	0	FALSE
"+-123"	N/A	N/A	0	FALSE

VAL_STRG 変換の例

この例は、次のように OUT 文字列を初期化した場合のものです。

"Current Temp = xxxxxxxxxx C"

「x」文字は、変換後の値に割り付けられる空白文字を表します。

データ タイプ	IN 値	Р	サイズ	FORMAT (W#16#)	PREC	OUT 文字列	ENO
UInt	123	16	10	0000	0	Current Temp = xxxxxxx123 C	TRUE
UInt	0	16	10	0000	2	Current Temp = xxxxxx0.00 C	TRUE
UDInt	12345678	16	10	0000	3	Current Temp = x12345.678 C	TRUE
UDInt	12345678	16	10	0001	3	Current Temp = x12345,678 C	TRUE
Int	123	16	10	0004	0	Current Temp = xxxxxx+123 C	TRUE
Int	-123	16	10	0004	0	Current Temp = xxxxxx-123 C	TRUE

データ タイプ	IN値	Р	サイズ	FORMAT (W#16#)	PREC	OUT 文字列	ENO
REAL	-0.00123	16	10	0004	4	Current Temp = xxx-0.0012 C	TRUE
REAL	-0.00123	16	10	0006	4	Current Temp = -1.2300E-3 C	TRUE
REAL	-INF	16	10	N/A	4	Current Temp = xxxxxx-INF C	FALS E
REAL	+INF	16	10	N/A	4	Current Temp = xxxxxx+INF C	FALS E
REAL	NaN	16	10	N/A	4	Current Temp = xxxxxxxNaN C	FALS E
UDInt	12345678	16	6	N/A	3	Current Temp = xxxxxxxxx C	FALS E

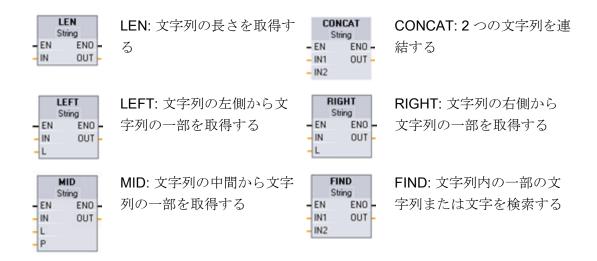
6.2.3.3 文字列操作命令

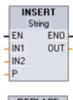
制御プログラムで、下記の文字列および文字命令を使用して、オペレータディスプレイおよびプロセスログ用のメッセージを作成することができます。

すべての文字列操作に共通のエラー

文字列操作命令の実行時に下記に示す不正または無効な文字列条件が発生したとき、その結果として ENO = 0 および Null 文字列出力が返されます。 特定の命令で発生するエラー条件を以下の命令操作の説明に示します。

ENO	条件	OUT
0	IN1 の現在の長さが IN1 の最大長を超えているか、IN2 の現在の長さが IN2 の最大長を超えている(不正な文字列) IN1、IN2、または OUT の最大長が、割り付けられているメモリ領域に適合していない	現在のデータ長 が 0 にセットさ れる
	IN1、IN2、または OUT の最大長が 0 または 255(不正な長 さ)	





INSERT: 文字列内に文字列 を挿入する



DELETE: 文字列から一部の 文字列を削除する



REPLACE: 文字列内の一部 の文字列を置換する

LEN 命令

パラメータ	パラメータ タイプ	データタイ プ	説明
IN	IN	文字列	入力文字列
OUT	OUT	UInt	IN 文字列内の有効な文字の数

LEN (文字列の長さ)は、文字列 IN の現在のデータ長を出力 OUT にセットします。 空の文字列のデータ長は 0 です。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	無効な文字列条件は存在しない	有効な文字列のデータ長

CONCAT 命令

パラメータ	パラメー タタイプ	データタイ プ	説明
IN1	IN	文字列	入力文字列 1
IN2	IN	文字列	入力文字列 2
OUT	OUT	文字列	連結された文字列(文字列 1 + 文字列 2)

CONCAT (文字列を連結する)は、String パラメータ IN1 と IN2 を 1 つに連結した文字 列を OUT にセットします。 連結された文字列の、String IN1 は左側の部分、String IN2 は右側の部分を構成します。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	連結の結果の文字列が、OUT 文字列の最	連結の結果の文字列が、OUTの最
	大長を超えている	大長になるまでコピーされる

LEFT 命令

パラメータ	パラメー タタイプ	データタイ プ	説明
IN	IN	文字列	入力文字列
L	IN	Int	作成する文字列の長さ(IN 文字列の左端から L 文字数)
OUT	OUT	文字列	出力文字列

LEFT (文字列の左側の一部)は、文字列パラメータ IN の最初の文字数 L の文字で構成された文字列を生成します。

- Lの値が IN 文字列の現在の長さよりも大きい場合は、IN 文字列全体を OUT に格納して返します。
- 入力が空の文字列である場合は、空の文字列を OUT に格納して返します。

この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	Lの値がO以下	現在のデータ長が 0 にセットされる
	コピーする文字列の長さ(L)が OUT 文字 列の最大長よりも大きい	コピーする文字列が OUT の最大長 になるまでコピーされる

RIGHT 命令

パラメータ	パラメー タタイプ	データタイ プ	説明
IN	IN	文字列	入力文字列
L	IN	Int	作成する文字列の長さ(IN 文字列の右端から L 文字数)
OUT	OUT	文字列	出力文字列

RIGHT (文字列の右側の一部)は、文字列の終りの L 文字の文字列を生成します。

- Lの値が IN 文字列の現在の長さよりも大きい場合は、IN 文字列全体をパラメータ OUT に格納して返します。
- 入力が空の文字列である場合は、空の文字列を OUT に格納して返します。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	L の値が 0 以下	現在のデータ長が 0 にセットされる
	コピーする文字列の長さ(L)が OUT 文字 列の最大長よりも大きい	コピーする文字列が OUT の最大長 になるまでコピーされる

MID 命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
IN	IN	文字列	入力文字列
L	IN	Int	作成する文字列の長さ(IN 文字列の文字位置 P から L 文字数)
Р	IN	Int	コピーする文字列の部分の最初の文字の位置: たとえば IN 文字列の最初の文字の位置は P= 1
OUT	OUT	文字列	出力文字列

MID (文字列の中間の一部)は、文字列の一部の部分の文字列を生成します。 文字列内の 文字の位置 P (P を含む)から L 文字数を抜き出した文字列を生成します。

LとPの和がStringパラメータINの現在の長さを超える場合は、文字位置Pから始まってIN文字列の終りまでに格納される文字数の文字列が返されます。この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	LまたはPの値がO以下	現在のデータ長が0にセットされ
	Pの値が IN の最大長よりも大きい	る
	コピーする文字列の長さ(L)が OUT 文字	Pの位置から始まる OUT の最大長
	列の最大長よりも大きい	になるまでの文字がコピーされる

DELETE 命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
IN	IN	文字列	入力文字列
L	IN	Int	削除する文字数
Р	IN	Int	削除する最初の文字の位置: IN 文字列の最初の文字の位置は P=1
OUT	OUT	文字列	出力文字列

DELETE (文字列の一部を削除する)は、文字列 IN 内の L 文字数の文字を削除します。 文字位置 P (P を含む)から L 文字削除した残りの文字列をパラメータ OUT に格納して返します。

- Lが0の場合は、入力文字列をOUTに格納して返します。
- L と P の和が入力文字列の長さよりも大きい場合は、文字列の最後まで削除します。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	P の値が IN の現在の長さよりも大きい L が 0 未満または P が 0 以下	IN の文字の削除は行われず、そのまま OUT にコピーされる 現在のデータ長が 0 にセットされ
	L かり木両または P かり以下	る しんじゅう 日本
	文字を削除した後の結果の文字列が、 OUT 文字列の最大長を超えている	結果の文字列が OUT の最大長になるまでコピーされる

INSERT

パラメー タ	パラメー タタイプ	データタ イプ	説明
IN1	IN	文字列	入力文字列 1
IN2	IN	文字列	入力文字列 2
Р	IN	Int	文字列 IN1 内における文字列 IN2 の挿入点の直前の文字位置。 文字列 IN1 内における最初の文字の位置は 1。
OUT	OUT	文字列	結果文字列

INSERT (文字列を挿入する)は、文字列 IN1 に文字列 IN2 を挿入します。 位置 P の直後の位置から挿入が開始されます。この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	Pの値が IN1 の長さよりも大きい	IN1 の最後の文字の直後に IN2 が 連結される
	Pの値が 0以下	現在のデータ長が 0 にセットされる
	挿入結果の文字列が、OUT 文字列の最大 長を超えている	結果の文字列が OUT の最大長になるまでコピーされる

REPLACE

パラメー タ	パラメー タタイプ	データタ イプ	説明
IN1	IN	文字列	入力文字列
IN2	IN	文字列	置換文字の文字列
L	IN	Int	置換する文字数
Р	IN	Int	置換する最初の文字の位置
OUT	OUT	文字列	結果文字列

REPLACE (文字列の一部を置換する)は、文字列パラメータ IN1 内の L 文字数の文字を置換します。 文字列パラメータ IN2 内の文字で、文字列 IN1 の文字位置 P (P も含む) から置換が開始されます。

- ・ パラメータ L の値が 0 の場合は、文字列 IN1 の文字位置 P に文字列 IN2 が挿入され、 文字列 IN1 内の文字の削除は行われません。
- **P** が **1** の場合は、文字列 **IN1** の最初の **L** 文字が文字列 **IN2** 内の文字に置換されます。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字
0	Pの値が IN1 の長さよりも大きい	IN1 の最後の文字の直後に IN2 が 連結される
	Pが IN1 内の文字位置であるが、IN1 内の残りの文字数が Lの値よりも小さい	IN1 の位置 P から残りの文字が IN2 に置換される
	Lが0未満またはPが0以下	現在のデータ長が 0 にセットされる
	置換結果の文字列が、OUT 文字列の最大 長を超えている	結果の文字列が OUT の最大長になるまでコピーされる

FIND

パラメー タ	パラメー タタイプ	データタ イプ	説明
IN1	IN	文字列	検索対象の文字列
IN2	IN	文字列	検索する文字列
OUT	OUT	Int	文字列 IN1 内で IN2 が最初に検索された文字位置

FIND (文字列の一部を検索する)は、文字列 IN1 内の、IN2 で指定した文字列または文字が存在する文字位置を取得します。 検索は左側から始まります。 IN2 文字列が存在する最初の文字位置を OUT にセットして返します。 文字列 IN1 内に文字列 IN2 が存在しない場合は 0 を返します。 この命令の条件コードを下の表に示します。

ENO	条件	OUT
1	エラーは検出されなかった	有効な文字位置
0	IN1 が IN2 よりも大きい	文字位置が 0 にセットされる

6.2.4 プログラム制御命令

6.2.4.1 スキャンサイクルウォッチドッグリセット命令



RE_TRIGR (スキャン時間ウォッチドッグ再トリガ)は、スキャンサイクルウォッチドッグタイマがエラーを生成する前に、最大サイクルタイムを延長するために使用します。

スキャンサイクル中にスキャンサイクルタイマを再スタートさせるには RE_TRIGR 命令を使用します。 前回の RE_TRIGR ファンクションの実行から、最大スキャンサイクルタイムを最大で 1 サイクルタイム延長する効果があります。

CPU は、RE_TRIGR 命令の使用を、OB1 およびサイクルプログラムから呼び出された ファンクションなどに制限します。 つまり、プログラムサイクル OB リスト内の OB から RE_TRIGR を呼び出した場合は、ウォッチドッグタイマがリセットされ、ENO = EN にセットされます。

RE_TRIGR がスタートアップ OB、割り込み OB、エラーOB から実行された場合は ENO = FALSE にセットされ、ウォッチドッグタイマはリセットされません。

PLC の最大サイクルタイムを設定する

PLC デバイス構成の[Cycle time]を使用して、最大スキャンサイクルタイムの値を設定することができます。

サタ	-イクルタイムモニ '	最小値	最大値	デフォルト値
最	大サイクルタイム	1 ms	6000 ms	150 ms

ウォッチドッグのタイムアウト

スキャンサイクルが完了する前に最大スキャンサイクルタイムが経過した場合には、エラーが生成されます。 ユーザープログラムにエラー処理プログラムブロック OB80 が存在する場合、PLC は OB80 を実行します。この OB80 に、独自の応答を作成するためのプログラムロジックを追加することができます。 OB80 が存在しない場合は、最初のタイムアウト条件は無視されます。

同じプログラムスキャン内で再び最大スキャンサイクルタイムのタイムアウトが発生した場合は(最大サイクルタイム値の 2 倍)、エラーがトリガされ、PLC は STOP モードに移行します。

STOP モードでは、PLC システムの通信およびシステム診断が行われている間、プログラムの実行が停止します。

6.2.4.2 スキャンサイクル停止命令



STP (PLC スキャンサイクルを停止する)は、PLC を STOP モードにします。 PLC が STOP モードのときは、プログラムの実行およびプロセスイメージからの物理的な更新は停止されます。

詳細については、以下を参照してください。 RUNモードからSTOPモードへの移行時の出力を設定する (ページ 60)

EN = TRUE の場合は、PLC は STOP モードに移行し、プログラムの実行が停止され、 ENO の状態が無意味になります。 そうでない場合は、EN = ENO = 0 にセットされます。

6.2.4.3 エラーを取得する命令

エラーを取得する命令によっては、プログラムブロックの実行エラーについての情報が提供されます。 プログラムブロックに GetError または GetErrorID 命令を追加して、プログラムブロック内でプログラムエラーを処理することができます。

GET_ERROR



GET_ERROR は、プログラムブロックの実行エラーが発生したことを示し、定義済みエラーデータ構造体に詳細なエラー情報を格納します。

パラメータ	データタイプ	説明
ERROR	ErrorStruct	エラーデータ構造体: この構造体の名前を変更することができますが、構造体内のメンバーの名前は変更
		できません。

ErrorStruct のデータ要	デー	説明
素	タタ	
	イプ	
ERROR_ID	WOR	エラーID
	D	
FLAGS	Byte	常に0にセットされる
REACTION	Byte	エラーに対する応答
		• 0 = 無視: 何も書き込まれない(書き込みエラー)
		• 1 = 代替: 入力値として 0 を使用(読み取りエラー)
		2 = 命令をスキップする
BLOCK_TYPE	Byte	エラーが発生したブロックの種類
		• 1 = OB
		• 2 = FC
		• 3 = FB
PAD_0	Byte	アラインメントのための内部充填バイト、通常は 0

ErrorStruct のデータ要	デー	説明
素	タタ	MAN
	イプ	
CODE_BLOCK_NUMB	UInt	エラーが発生したブロックの番号
ER		
ADDRESS	UDInt	エラーを生成した命令の内部メモリロケーション
MODE	Byte	どのように残りのフィールドを解釈するかを示す内
		部マッピングで、STEP 7 Basic が使用する
PAD_1	Byte	アラインメントのための内部充填バイトで未使用、
		通常は0
OPERAND_NUMBER	UInt	内部命令オペランド番号
POINTER_NUMBER_	UInt	(A) 内部命令ポインタのロケーション
LOCATION		
SLOT_NUMBER_SCO	UInt	(B) 内部メモリストレージのロケーション
PE		
AREA	Byte	(C) エラーが発生した時に参照されるメモリ領域
		• L:16#40 – 4E、86、87、8E、8F、C0 – CE
		• I: 16#81
		• Q: 16#82
		• M: 16#83
		• DB: 16#84、85、8A、8B
PAD_2	Byte	アラインメントのための内部充填バイトで未使用、
		通常は0
DB_NUMBER	UInt	(D) DB エラーが発生したときに参照される DB、そ
		うでない場合は0
OFFSET	UDInt	(E) エラーが発生したときに参照されるビットオフセ
		ット(例: 12 = バイト 1、ビット 4)

GET_ERR_ID



GET_ERR_ID は、プログラムブロックエラーが発生したことを示し、そのエラーID (識別コード)を報告します。

パラメータ	データタイ プ	説明
ID	WORD	ErrorStruct の ERROR_ID メンバーに格納するエラーID の値

ERROR_ID Hexadecimal	ERROR_ID Decimal	プログラムブロック実行エラー	
2503	9475	初期化されていないポインタエラー	
2522	9506	オペランド範囲外の読み取りエラー	
2523	9507	オペランド範囲外への書き込みエラー	
2524	9508	無効な領域の読み取りエラー	
2525	9509	無効な領域への書き込みエラー	
2528	9512	データアラインメント読み取りエラー(ビットアラインメント不正)	
2529	9513	データアラインメント書き込みエラー(ビットアラインメント不正)	
2530	9520	DB が書き込み保護	
253A	9530	グローバル DB が存在しない	
253C	9532	不正なバージョンまたは FC が存在しない	
253D	9533	命令が存在しない	
253E	9534	不正なバージョンまたは FB が存在しない	
253F	9535	命令が存在しない	
2575	9589	プログラムのネスト深度エラー	
2576	9590	ローカルデータの割り付けエラー	
2942	10562	物理入力ポイントが存在しない	
2943	10563	物理出力ポイントが存在しない	

操作

デフォルトでは、CPU はブロック実行エラーに応答して、診断バッファにエラーを記録し、STOP モードに移行します。 ただし、プログラムブロック内に 1 つ以上のGET_ERROR または ERR_ID 命令を配置した場合、このブロックはブロック内のエラーを処理するようにセットされます。 この場合、CPU は STOP モードに移行せず、診断バッファにエラーログを記録しません。 エラー情報が、GET_ERROR またはGET_ERR_ID 命令の出力として報告されます。 GET_ERROR 命令を使用して、詳細なエラー情報を読み取ることができます。 さらに GET_ERR_ID 命令を使用して、エラーID だけを読み取ることもできます。 通常は、最初のエラーが最も重要で、それ以降のエラーは最初のエラーの結果に過ぎません。

ブロック内の GET_ERROR または ERR_ID 命令の最初の実行で、ブロックの実行時に 検出された最初のエラーが返されます。 このエラーは、ブロックの始まりから GET_ERROR または ERR_ID 命令が実行されるまでのいずれの場所でも発生する可能 性があります。 GET_ERROR または ERR_ID 命令の次のどちらかの実行では、前の GET_ERROR または ERR_ID 命令の実行以降の最初のエラーが返されます。エラーの 履歴は保存されません。どちらの命令を実行しても、PLC システムが次のエラーを見 つけるようにします。

GET_ERROR 命令で使用する ErrorStruct データタイプをデータブロックエディタで追加し、これらの値にプログラムロジックからアクセスすることができます。 データタイプドロップダウンリストで、この構造体に追加する ErrorStruct を選択します。 固有の名前を使用することによって、複数の ErrorStruct を作成することができます。 ただし、ErrorStruct のメンバーの名前を変更することはできません。

ENO によって示されるエラー条件

EN = TRUE のとき、GET_ERROR または GET_ERR_ID を実行すると、

- ENO = TRUE は、プログラムブロック実行エラーが発生してエラーデータが存在することを示します。
- ENO = FALSE は、プログラムブロック実行エラーが発生しなかったことを示します。

エラーが発生した場合に実行するエラー応答プログラムロジックを ENO に接続することができます。 エラーが存在する場合は、出力パラメータにエラーデータが格納され、プログラムからアクセスすることができます。

実行中のブロック(呼び出されたブロック)から呼び出し元のブロックにエラー情報を送るために、GET_ERROR および GET_ERR_ID を使用することができます。 呼び出されたブロックプログラムの最後のネットワーク内にこの命令を配置し、そのブロックの最終的な実行ステータスを報告するようにします。

6.2.5 通信命令

6.2.5.1 イーサネット通信を開く

自動接続/切断(TSEND_C / RCV_C)を使用してイーサネット通信を開く

注記

TSEND_C命令およびTRCV_C命令の処理にかかる時間は不定です。各スキャンサイクルでこれらの命令を確実に処理するために、プログラムサイクル OB やプログラムサイクルスキャンから呼び出したプログラムブロックなど、常にメインプログラムサイクルスキャンから呼び出すようにします。これらの命令は、ハードウェア割り込み OB、遅延割り込み OB、周期割り込み OB、エラー割り込み OB、またはスタートアップ OB から呼び出さないでください。

これらの命令によるデータ転送の詳細については、データの一貫性 (ページ 114)に関するセクションを参照してください。

TSEND_C の説明

TSEND_C は、相手ステーションとの TCP または ISO on TCP の通信接続を確立し、データを送信して、接続を終了することができます。 セットアップおよび確立された接続は、CPU によって自動的に維持され、モニタされます。 TSEND_C は、TCON、TDISCON、および TSEND の機能を 1 つにまとめた命令です。

TSEND C命令を使用して送信できるデータの最小サイズは1バイトです。

注記

LEN パラメータのデフォルト設定(LEN = 0)では、送信するデータ長を決定するために DATA パラメータを使用します。 TSEND_C 命令によって送信される DATA は、 TRCV_C 命令の DATA パラメータと同じサイズでなければなりません。

TSEND_C命令の動作は次のようになります。

- 接続を確立するために、CONT = 1 を使用して TSEND C を実行します。
- 接続が正常に確立したら、TSEND_C は 1 サイクルの DONE パラメータをセットします。
- 通信接続を終了するために、CONT = 0 を使用して TSEND_C を実行します。接続 が直ちに切断されます。 これは受信側ステーションにも影響を与えます。 受信側 での接続が閉じて、受信バッファ内のデータが失われる可能性があります。
- 確立された接続を通じてデータを送信するために、REQ の立ち上がりエッジを使用して TSEND_C を実行します。 正常に送信が行われたら、TSEND_C は 1 サイクルの時間 DONE パラメータをセットします。
- 接続を確立してデータを送信するために、CONT =1 および REQ = 1 を使用して TSEND_C を実行します。正常に送信が行われた後、TSEND_C は 1 サイクルの DONE パラメータをセットします。

TRCV_C

TRCV_C は、パートナーCPU との TCP または ISO on TCP の通信接続を確立し、データを受信して接続を終了することができます。 セットアップおよび確立された接続は、CPU によって自動的に維持され、モニタされます。 TRCV_C 命令は、TCON、TDISCON、TRCV 命令のファンクションを組み合わせたものです。

TRCV_C 命令を使用して受信できるデータの最小サイズは 1 バイトです。 TRCV_C 命令では、ブールデータおよびブール配列はサポートされません。

注記

LEN パラメータのデフォルト設定(LEN = 0)では、送信するデータ長を決定するために DATA パラメータを使用します。 TSEND_C 命令によって送信される DATA は、 TRCV_C 命令の DATA パラメータと同じサイズでなければなりません。

TRCV_C 命令の動作は次のようになります。

- 接続を確立するために、CONT = 1 を使用して TRCV_C を実行します。
- データを受信するために、 $EN_R = 1$ を使用して $TRCV_C$ を実行します。 $TRCV_C$ は、 $EN_R = 1$ および CONT = 1 のときに、連続してデータを受信します。
- 通信接続を終了するために、パラメータ CONT = 0 を使用して TRCV_C を実行します。接続が直ちに切断されます。

受信モード

TRCV_C は、TRCV 命令と同じ受信サービスモードを処理します。 どのようにして受信領域にデータが入力されるのかを下の表に示します。

プロトコルの種類	受信領域へのデータの入力	パラメータ connection_type
TCP	指定された長さのデータ受信	B#16#11
ISO on TCP	プロトコル制御	B#16#12

注記

TSEND_C の処理は非同期のため、DONE パラメータまたは ERROR パラメータが TRUE である間は、送信側領域でデータの一貫性を保持する必要があります。

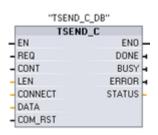
TSEND_C の場合、パラメータ DONE が TRUE である状態は、データが正常に送信されたことを意味します。 相手 CPU の接続が受信バッファを実際に読み取ったことを意味しているわけではありません。

TRCV_C の処理は非同期で行われるため、受信側領域のデータは DONE = 1 のときにのみ一貫性があります。

パラメータ BUSY、DONE および ERROR の関係を下の表に示します。

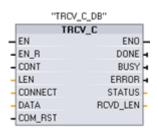
BUSY	DONE	ERROR	説明
TRUE	対象外	対象外	ジョブが処理されています。
FALSE	TRUE	FALSE	ジョブが正常に終了しました。
FALSE	FALSE	TRUE	ジョブにエラーが発生して終了しました。 エラーの原
			因は STATUS パラメータで見つかります。
FALSE	FALSE	FALSE	新しいジョブが割り付けられませんでした。

TSEND_C のパラメータ



パラメータ	パラメータ	データタ	説明
	タイプ	イプ	
REQ	INPUT	BOOL	制御パラメータ REQ が、立ち上がりエッジ
			で、CONNECT で表された接続を使用して 送信
			ジョブを開始します。
CONT	INPUT	BOOL	• 0: 切断
			• 1 : 接続を確立して保持
LEN	INPUT	Int	送信する最大バイト数 (デフォルト = 0、DATA
			パラメータによって送信するデータ長を決定す
			る)。
CONNECT	IN_OUT	TCON-	接続記述へのポインタ
		Param	
DATA	IN_OUT	Variant	送信領域、アドレスおよび送信するデータ長を
			格納
COM_RST	IN_OUT	BOOL	• 1: ファンクションブロックの完全なリスター
			ト、既存の接続を終了する。
DONE	OUTPUT	BOOL	• 0: ジョブは起動していないか、または実行中
			• 1: ジョブは正常に実行されました。
BUSY	OUTPUT	BOOL	• 0: ジョブが完了しています
			• 1: ジョブはまだ完了していません。 ジョブ
			を新たにトリガできません。
ERROR	OUTPUT	BOOL	• 1: 処理中にエラーが発生 エラータイプにつ
			いての詳しい情報が STATUS に表示されま
			す。
STATUS	OUTPUT	WORD	エラー情報

TRCV_C のパラメータ



パラメータ	パラメータ	データタ	説明	
	タイプ	イプ		
EN_R	IN	BOOL	受信が有効化された制御パラメータ。 EN_R = 1	
			のとき、TRCV_C 受信準備完了 受信ジョブが処理	
			されます。	
CONT	IN	BOOL	制御パラメータ CONT	
			• 0: 切断	
			• 1:接続を確立して保持	
LEN	IN	Int	受信領域のバイト単位の長さ (デフォルト = 0、	
			DATA パラメータによって送信するデータ長を決	
			定する)。	
CONNECT	IN_OUT	TCON-	接続記述へのポインタ	
		Param		
DATA	IN_OUT	Variant	受信領域に受信したデータの先頭アドレスと最大	
			長が格納される。	
COM_RST	IN_OUT	BOOL	• 1: ファンクションブロックの完全なリスター	
			ト、既存の接続を終了する。	
DONE	OUT	BOOL	• 0: ジョブは起動していないか、または実行中	
			• 1: ジョブは正常に実行されました。	
BUSY	OUT	BOOL	• 0: ジョブが完了しています	
			• 1: ジョブはまだ完了していません。 ジョブを	
			新たにトリガできません。	
ERROR	OUT	BOOL	• 1: 処理中にエラーが発生 エラータイプについ	
			ての詳しい情報が STATUS に表示されます。	
STATUS	OUT	WORD	エラー情報	
RCVD_LEN	OUT	Int	バイト単位で実際に受信したデータの量	

パラメータ Error および Status

ERROR	STATUS (W#16#)	説明			
0	0000	ジョブは正常に実行されました。			
0	7000	アクティブなジョブ処理はありません。			
0	7001	ジョブの処理を開始し、接続を確立して、接続相手を待機する			
0	7002	送受信中のデータ			
0	7003	接続終了			
0	7004	接続を確立してモニタ、処理中のジョブは存在しない			
1	8085	LEN パラメータが、最大許容値よりも大きくなっています			
1	8086	CONNECT パラメータの値が、許容範囲外			
1	8087	最大接続数に達している。これ以上の追加接続はできません。			
1	8088	LEN パラメータの値が DATA で指定したメモリ領域よりも大きい。受信メモリ領域が小さ過ぎる			
1	8089	パラメータ CONNECT パラメータがデータブロックを指定していません。			
1	8091	最大ネスト深度を超過			
1	809A	CONNECT パラメータが、接続記述の長さに適合しないフィールドを指している			
1	809B	接続設定の local_device_id が CPU と一致しません。			
1	80A1	通信エラー: ・ 指定された接続は、まだ確立されていませんでした。 ・ 指定した接続は終了中で、この接続を通じての転送は不可 ・ インターフェースの再初期化中			
1	80A3	存在しない接続を終了しようとしています			
1	80A4	リモートの接続相手のIPアドレスが無効。 リモートの接続相手のIPアドレスがローカル接続相手のIPアドレスと同じなど。			
1	80A7	通信エラー: TCON が完了する前に TDISCON を呼び出した(ID によって参照されている接続を TDISCON が先に完了する必要 がある)			
1	80B2	CONNECT パラメータが、キーワード UNLINKED を使用して 生成されたデータブロックを指している			

ERROR	STATUS	説明
	(W#16#)	
1	80B3	パラメータの不整合:
		• 接続の詳細でエラーが発生しました
		• ローカルポート(パラメータ local_tsap_id)は、既に別の接続
		設定にあります
		• 接続設定の ID がパラメータで指定された ID と異なります
1	80B4	ISO on TCP を使用して受動接続を確立すると(connection_type = B#16#12)、条件コード 80B4 で、入力した TSAP が次のアドレス要件と一致しないことが警告されます。 ・ ローカル TSAP の長さが 2 で、最初のバイトの TSAP ID の値が E0 または E1 の場合(16 進数値)、2 番目のバイトは 00 または 01 でなければなりません。 ・ ローカル TSAP の長さが 3 以上で、最初のバイトの TSAP ID 値が E0 または E1 の場合(16 進数)、2 番目のバイトは 00 または 01 で、他のすべてのバイトが有効な ASCII 文字でなければなりません。 ・ ローカル TSAP の長さが 3 以上で、TSAP ID の最初のバイトの値が E0 でも E1 でもない場合(16 進数値)、TSAP ID のすべてのバイトが有効な ASCII 文字でなければなりません。 有効な ASCII 文字は 20~7E (16 進数値)のバイト値になりま
1	80C3	す。 すべての接続リソースが使用中です。
1	80C4	一時的な通信エラー:
	0004	●・現時点での接続が確立できません
		インターフェースが新しいパラメータを受信しています
		この構成の接続は TDISCON によって削除中
1	8722	CONNECT パラメータ: ソース領域無効: DB 内に領域が存在しない
1	873A	CONNECT パラメータ: 接続記述にアクセスできない(DB が使用できないなど)
1	877F	CONNECT パラメータ: 無効な ANY 参照などの内部エラー

接続/切断コントロールを使ってイーサネット通信を開きます

注記

TCON、TDISCON、TSEND、TRCV命令の処理には、不確定の時間数がかかる可能性があります。各スキャンサイクルでこれらの命令を確実に処理するために、プログラムサイクル OB やプログラムサイクルスキャンから呼び出したプログラムブロックなど、常にメインプログラムサイクルスキャンから呼び出すようにします。 これらの命令は、ハードウェア割り込み OB、遅延割り込み OB、周期割り込み OB、エラー割り込み OB、またはスタートアップ OB から呼び出さないでください。

TCP および ISO on TCP を使用するイーサネット通信

以下のプログラム命令は通信プロセスを制御します。

- TCON は接続を行います。
- TSEND と TRCV はデータを送受信します。
- TDISCON は通信を切断します。

TSEND命令とTRCV命令を使用して送受信できるデータの最小サイズは、1 バイトです。TRCV命令は、ブールデータまたはブール配列の送信をサポートしません。 これらの命令によるデータ転送の詳細については、データの一貫性 (ページ 114)に関するセクションを参照してください。

注記

LEN パラメータのデフォルト設定(LEN = 0)では、送信するデータ長を決定するために DATA パラメータを使用します。 TSEND 命令によって送信する DATA のサイズを TRCV 命令の DATA パラメータと同じサイズにしてください。

両方の通信パートナーが通信接続をセットアップおよび確立するために TCON 命令を呼び出します。 パラメータを使用して、アクティブ通信とパッシブ通信のエンドポイントパートナーを指定します。 セットアップおよび確立された接続は、CPU によって自動的に維持され、モニタされます。

ライン切断やリモート通信パートナーによって接続が終了した場合、送信側のパートナーは、構成された接続の再確立を試みます。 TCON を再実行する必要はありません。

TDISCON 命令が実行されたとき、または CPU が STOP モードになったときは、既存の接続が終了し、設定された接続が削除されます。 接続を設定、確立するには、TCONを再実行する必要があります。

ファンクションの説明

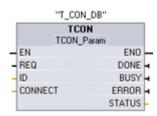
TCON、TDISCON、TSEND、TRCV は同時に動作します。つまり、ジョブの処理は複数の命令にわたって実行されます。

たとえば、パラメータ REQ = 1 で TCON 命令を実行することで、接続の設定と確立を 開始するためにジョブを開始します。次に追加で TCON を実行して、ジョブの進捗を モニタし、パラメータ DONE を使ってジョブの完了をテストします。

以下の表は、BUSY、DONE、ERRORの関係を示しています。 この表を使用して現在 のジョブステータスを判別します。

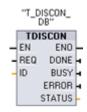
BUSY	DONE	ERROR	説明
TRUE	対象外	対象外	ジョブが処理されています。
FALSE	TRUE	FALSE	ジョブが正常に終了しました。
FALSE	FALSE	TRUE	ジョブにエラーが発生して終了しました。 エラーの原
			因は STATUS パラメータで見つかります。
FALSE	FALSE	FALSE	新しいジョブが割り付けられませんでした。

TCON



パラメータ	パラメー タタイプ	データタイ プ	説明	
REQ	IN	Bool	Bool 制御パラメータ REQUEST によって、ID で指定された接続を確立するジョブが開始します。ジョブは REQUEST の立ち上がりエッジで開始します。	
ID	IN	CONN_OUC リモートパートナーに対して、またはユーザープログラムとオペレーティングシステムの通信レイヤーとの間で確立される接続への参照。 ID はローカル接続の設定内容の関連パラメータ ID と同一でなければなりません。値の範囲 W#16#0001~W#16#0FFF		
CONNECT	IN_OUT	TCON- Param	接続記述へのポインタ	
DONE	OUT	Bool	ステータスパラメータ DONE: • 0: ジョブは起動していないか、または実行中 • 1: ジョブは正常に実行されました。	
BUSY	OUT	Bool	BUSY=1: ジョブはまだ完了していません。 BUSY=0: ジョブが完了しました。	
ERROR	OUT	Bool	ステータスパラメータ ERROR です。 ERROR = 1: ジョブ処理時に生じたエラー: エラータイプについての詳しい情報が STATUS に表示されます。	
STATUS	OUT	Word	ステータスパラメータ STATUS: エラー情報	

TDISCON



TCP および ISO on TCP: TDISCON は CPU から通信パートナーへの通信接続を終了します。

パラメー タ	パラメータ タイプ	データタイ プ	説明	
REQ	IN	Bool 制御パラメータ REQUEST によって、I 定された接続を確立するジョブが開始しジョブは REQUEST の立ち上がりエッサムします。		
ID	IN	CONN_OU フェートパートナーに対して、またはユーザー プログラムとオペレーティングシステムの通信 レベルとの間で終了する接続への参照。 ID はローカル接続の設定内容の関連パラメータ ID と同一でなければなりません。 値の範囲 W#16#0001~W#16#0FFF		
DONE	OUT	Bool ステータスパラメータ DONE:		
BUSY	OUT	Bool	BUSY=1: ジョブはまだ完了していません。 BUSY=0: ジョブが完了しました。	
ERROR	OUT	Bool	Bool ERROR = 1: 処理中にエラーが発生	
STATUS	OUT	Word	エラーコード:	

TSEND

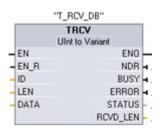
"T_SEND_DB_

		'	
		TSEND	1
		UInt to Variant	ı
-	EN	ENO	H
-	REQ	DONE	H
-	ID	BUSY	H
-	LEN	ERROR	k
-	DATA	STATUS	ŀ

パラメー タ	パラメータ タイプ	データタイ プ	説明	
REQ	IN	Bool	制御パラメータ REQUEST の立ち上がりエッジで送信ジョブが開始します。	
		DATA および LEN で指定された領域から、データが転送されます。		
ID	IN	CONN_OU 関連付けられた接続への参照。 ID はローカル C (Word) 接続の設定内容の関連パラメータ ID と同一でなければなりません。 値の範囲 W#16#0001~W#16#0FFF		
LEN	IN	Int	ジョブで送信される最大バイト数	
DATA	IN_OUT	Variant	データ送信領域を指すポインタです。 送信側 領域、アドレスとデータ長が含まれます。 ア ドレスは以下を参照します。 ・ プロセスイメージ入力テーブル ・ プロセスイメージ出力テーブル ・ ビットメモリ ・ データブロック	
DONE	OUT	Bool	ステータスパラメータ DONE: • 0: ジョブは起動していないか、または実行中 • 1: ジョブは正常に実行されました。	
BUSY	OUT	 Bool BUSY=1: ジョブはまだ完了していません。 ジョブを新たにトリガできません。 BUSY=0: ジョブが完了しました。 		

パラメー タ	パラメータ タイプ	データタイ プ	説明
ERROR	OUT	Bool	ステータスパラメータ ERROR です。 ERROR = 1: 処理中にエラーが発生 エラータ イプについての詳しい情報が STATUS に表示 されます。
STATUS	OUT	Word	ステータスパラメータ STATUS: エラー情報

TRCV



パラメータ	パラメー タタイプ	データタイ プ	説明
EN_R	IN	Bool 受信が有効化された制御パラメータ。 EN_R = 1 のとき、TRCV は受信可能です。 受信ジョブが処理中です。	
ID	IN	CONN_OU C (Word)	関連付けられた接続への参照。 ID はローカル 接続の設定内容の関連パラメータ ID と同一で なければなりません。 値の範囲 W#16#0001~W#16#0FFF
LEN	IN	Int 受信領域のバイト単位の長さ(デフォルト= 0、 この場合 DATA パラメータが受信されるデー タ長を指定します)。	
DATA	IN_OUT	Variant	受信データへのポインタです。 アドレスとデータ長を含む受信領域。 アドレスは以下を参照します。 ・ プロセスイメージ入力テーブル ・ プロセスイメージ出力テーブル ・ ビットメモリ ・ データブロック
NDR	OUT	Bool	ステータスパラメータ NDR: • NDR = 0: ジョブは起動していないか、または実行中 • NDR = 1: ジョブが正常に終了しました。

パラメータ	パラメー タタイプ	データタイ プ	説明
BUSY	OUT	Bool	 BUSY=1: ジョブはまだ完了していません。 ジョブを新たにトリガできません。 BUSY=0: ジョブが完了しました。
ERROR	OUT	Bool	ERROR=1: 処理中にエラーが発生 エラータイプについての詳しい情報が STATUS に表示されます。
STATUS	OUT	Word	エラー情報
RCVD_LE N	OUT	Int	バイト単位で実際に受信したデータの量

受信領域

TRCV 命令は、受信データを次の2つの変数で指定された受信領域に書き込みます。

- 受信領域の開始を示すポインタ
- 受信領域の長さ

注記

LEN パラメータのデフォルト設定(LEN = 0)では、送信するデータ長を決定するため に DATA パラメータを使用します。 TSEND 命令によって送信する DATA のサイズ を TRCV 命令の DATA パラメータと同じサイズにしてください。

次の表には、TRCV により受信データがどのように受信領域に入るかが示されています。

プロトコルの種類	受信領域へのデータ入力	パラメータ接続タイプ
TCP	指定された長さのデータ受信	B#16#11
ISO on TCP	プロトコル制御	B#16#12

ジョブのすべてのデータを受信すると、TRCV によって直ちにそれが受信領域に転送され、NDR が 1 にセットされます。

TCON の条件コード

ERRO R	STATUS (W#16#	説明	
0	0000	接続が正常に確立しました。	
0	7000	アクティブなジョブ処理はありません。	
0	7001	ジョブ処理を開始して、接続を確立します。	
0	7002	後続呼び出し(REQ は対象外)。接続を確立しています。	
1	8086	ID パラメータが許容範囲内にありません。	
1	8087	最大接続数に達している。これ以上の追加接続はできません。	
1	809B	接続設定の local_device_id が CPU と一致しません。	
1	80A1	接続またはポートは既にユーザーによって占有されています。	
1	80A2	ローカルまたはリモートポートがシステムによって占有されてい ます。	
1	80A3	既存の接続の再確立を試行しています。	
1	80A4	リモート接続エンドポイントの IP アドレスが不正です。ローカル IP アドレスに一致している可能性があります。	
1	80A7	通信エラー: TCON の完了前に TDISCON が実行されました。 TDISCON は ID で参照される接続を最初に完全に終了する必要があります。	
1	80B3	パラメータ割り付けに一貫性がありません。 エラーコード W#16#80A0~W#16#80A2、W#16#80A4、W#16#80B4~ W#16#80B9 のグループエラーです。	

ERRO R	STATUS (W#16# .)	説明	
1	80B4	ISO on TCP を使用して受動接続を確立すると(connection_type = B#16#12)、条件コード 80B4 で、入力した TSAP が次のアドレス要件と一致しないことが警告されます。	
		 ローカル TSAP の長さが 2 で、最初のバイトの TSAP ID の値が E0 または E1 の場合(16 進数値)、2 番目のバイトは 00 または 01 でなければなりません。 	
		• ローカル TSAP の長さが 3 以上で、最初のバイトの TSAP ID 値が E0 または E1 の場合(16 進数)、2 番目のバイトは 00 または 01 で、他のすべてのバイトが有効な ASCII 文字でなければなりません。	
		• ローカル TSAP の長さが 3 以上で、TSAP ID の最初のバイト の値が E0 でも E1 でもない場合(16 進数値)、TSAP ID のすべ てのバイトが有効な ASCII 文字でなければなりません。	
		有効な ASCII 文字は 20~7E (16 進数値)のバイト値になります。	
1	80B5	active_est のエラーパラメータ	
1	80B6	パラメータ connection_type でのパラメータ割り付けエラー	
1	80B7	次のいずれかのパラメータでのエラーです。 block_length、 local_tsap_id_len、rem_subnet_id_len、rem_staddr_len、 rem_tsap_id_len、next_staddr_len	
1	80B8	ローカル接続記述のパラメータとパラメータ ID が違います。	
1	80C3	すべての接続リソースが使用中です。	
1	80C4	一時的な通信エラー:・ 現時点での接続が確立できません。	
		インターフェースが新しいパラメータを受信しています。設定された接続は、現在 TDISCON によって削除されています。	

TDISCON の条件コード

ERRO R	STATUS (W#16#	説明	
0	0000	接続が正常に終了しました。	
0	7000	アクティブなジョブ処理はありません。	
0	7001	ジョブ処理を開始して、接続を終了します。	
0	7002	後続の呼び出し(REQ は対象外)。接続を終了しています。	
1	8086	ID パラメータが使用できるアドレス範囲になっていません。	
1	80A3	存在しない接続を終了しようとしています。	
1	80C4	一時的な通信エラー: インターフェースが新規パラメータを受信中、または接続が現在確立中です。	

TSEND の条件コード

ERRO R	STATUS (W#16#	説明	
0	0000	ジョブの送信がエラーなしで完了	
0	7000	アクティブなジョブ処理はありません。	
0	7001	ジョブ処理を開始して、データを送信します。 この処理中、オペレーティングシステムが DATA 送信領域内のデータにアクセスします。	
0	7002	追加呼び出し(REQ は対象外)。ジョブが処理されています。 この 処理中、オペレーティングシステムが DATA 送信領域内のデータ にアクセスします。	
1	8085	LEN パラメータが、最大許容値よりも大きくなっています。	
1	8086	ID パラメータが使用できるアドレス範囲になっていません。	
1	8088	LEN パラメータが、DATA で指定されているメモリ領域よりも大きくなっています。	

ERRO R	STATUS (W#16# .)	説明
1	80A1	通信エラー:
		• 指定された接続は、まだ確立されていませんでした。
		• 指定された接続は、現在終了中です。 この接続を介した転送
		はできません。
		• インターフェースの再初期化中。
1	80C3	内部のリソース不足 この ID のブロックは、既に異なる優先度ク
		ラスで処理されています。
1	80C4	一時的な通信エラー:
		• 現在、通信パートナーへの接続を確立できません。
		• インターフェースが新規パラメータを受信中、または接続が現
		在確立中です。

TRCV の条件コード

ERRO R	STATUS (W#16# .)	説明	
0	0000	新しいデータを受信しました。 受信されたデータの現在の長さは RCVD_LEN に表示されます。	
0	7000	ブロックは受信の用意ができていません。	
0	7001	ブロックは受信の用意ができています。受信ジョブが実行されま した。	
0	7002	後続呼び出し、受信ジョブが処理されています。 この処理中にデータは受信領域に書き込まれます。このため、エラーが発生すると受信領域のデータに一貫性がなくなることがあります。	
1	8085	LEN パラメータが最大許容値よりも大きいか、または最初の呼び 出しを行ってから、ユーザーにより LEN または DATA パラメー タが変更されました。	
1	8086	ID パラメータが使用できるアドレス範囲になっていません。	
1	8088	受信領域が小さ過ぎます。 LEN の値が、DATA によって指定した 受信領域よりも大きくなっています。	

ERRO R	STATUS (W#16# .)	説明		
1	80A1	通信エラー:		
		• 指定された接続が、まだ確立されていません。		
		• 指定された接続は、現在終了中です。 この接続を経由した受信ジョブを行うことはできません。		
		インターフェースが新しいパラメータを受信しています。		
1	80C3	内部のリソース不足 この ID のブロックは、既に異なる優先度クラスで処理されています。		
1	80C4	一時的な通信エラー:		
		• 現在、通信パートナーへの接続を確立できません。		
		• インターフェースが新規パラメータ設定を受信中、または接続が現在確立中です。		

6.2.5.2 ポイントツーポイント命令

「ポイントツーポイント(PtP)」の章 (ページ 331)には、ポイントツーポイント命令と 通信モジュールに関する詳細情報が記載されています。

6.2.6 割り込み命令

6.2.6.1 アタッチおよびデタッチ命令





ATTACH 命令と DETACH 命令を使用して、割り込みイベント駆動のサブプログラムを有効または無効にすることができます。

- ATTACH を使用して、ハードウェア割り込みイベントの ための OB サブプログラム実行の割り込みを有効にする ことができます。
- DETACH を使用して、ハードウェア割り込みイベントの ための OB サブプログラム実行の割り込みを無効にする ことができます。

パラメータ	パラメータ タイプ	データタ イプ	説明
OB_NR	IN	Int	オーガニゼーションブロック識別子: [Add new block]機能を使用して作成された、利用可能なハードウェア割り込み OB から選択します。 このパラメータフィールドをダブルクリックしてから、ヘルパーアイコンをクリックして利用可能な OB を表示します。
EVENT	IN	DWord	イベント識別子: デジタル入力または高速カウンタ用に PLC デバイスコンフィグレーションで有効にされ た、利用可能なハードウェア割り込みイベン トから選択します。 このパラメータフィー ルドをダブルクリックしてから、ヘルパーア イコンをクリックして利用可能なイベントを 表示します。

パラメータ	パラメータ タイプ	データタ イプ	説明
ADD (ATTACH の み)	IN	Bool	ADD = 0 (デフォルト): このイベントは、この OB の前のすべてのイベントのアタッチメントを置換します。 ADD = 1: このイベントは、この OB の前のイベントのアタッチメントに追加されました。
RET_VAL	OUT	Int	実行条件コード

ハードウェア割り込みイベント

CPU によって、以下のハードウェア割り込みイベントがサポートされます。

- 立ち上がりエッジイベント (すべてのビルトイン CPU デジタル入力と、任意のシグナルボードデジタル入力)
 - 立ち上がりエッジは、接続されたフィールドデバイスから入力への信号の変化に応じて、デジタル入力の OFF から ON への移行時に発生します。
- 立ち下がりエッジイベント (すべてのビルトイン CPU デジタル入力と、任意のシグ ナルボードデジタル入力)
 - 立ち下がりエッジは、デジタル入力の ON から OFF への移行時に発生します。
- 高速カウンタ(HSC)の現在の値=基準値(CV = RV)イベント(HSC 1~6)
 - 現在のカウントが設定された基準値に一致するときに、HSC の CV = RV 割り込みが発生します。
- HSC 方向変更イベント(HSC 1~6)
 - 方向変更イベントは、HSC が増加から減少、または減少から増加への変更を検出 された場合に発生します。
- HSC 外部リセットイベント(HSC 1~6)
 - 特定の HSC モードを使用して、デジタル入力の HSC カウント値をゼロにリセットするため使用する外部リセットとして割り付けることができます。 HSC などの入力が OFF から ON に移行すると、外部リセットイベントが発生します。

デバイスコンフィグレーションでのハードウェア割り込みイベントの有効化

ハードウェア割り込みは、デバイスコンフィグレーション内で有効にする必要があります。 コンフィグレーションまたはランタイム中にこのイベントをアタッチする場合は、デジタル入力チャンネルまたは HSC のデバイスコンフィグレーションで有効にするチェックボックスをオンにする必要があります。

PLC デバイスコンフィグレーションでのチェックボックスのオプションは次の通りです。

- デジタル入力
 - 立ち上がりエッジ検出の有効化
 - 立ち下がりエッジ検出の有効化
- - 高速カウンタ使用の有効化
 - カウンタ値の割り込み生成=基準値カウント
 - 外部リセットイベントの割り込み生成
 - 方向変更イベントの割り込み生成

新しいハードウェア割り込み OB プログラムブロックのプログラムへの追加

デフォルトでは、イベントが最初に有効になると OB は割り付けられません。 これは [HW interrupt:] デバイスコンフィグレーション[<not connected>]ラベルで表示されます。 ハードウェア割り込み OB だけが、ハードウェア割り込みイベントにアタッチできます。 すべての既存のハードウェア割り込み OB は、[HW interrupt:]ドロップダウンリストに 表示されます。 ドロップダウンリスト。 OB が表示されない場合、次のように [Hardware interrupt]タイプの OB を作成する必要があります。 プロジェクトツリーの [Program blocks]分岐で次の操作を実行します。

- 1. [Add new block]をダブルクリックし、[Organization block (OB)]、[Hardware interrupt]の順に選択します。
- 2. オプションで、OB の名前を変更し、プログラム言語(LAD または FBD)を選択して、ブロック番号を選択できます(手動に切り替え、提案されるブロック番号とは異なる番号を選択します)。
- 3. OB を編集し、イベント発生時に実行させるプログラム応答を追加します。 この OB から FC と FB を呼び出せます(ネストの深さは 4 までです)。

OB_NR パラメータ

すべての既存のハードウェア割り込み OB 名は、デバイスコンフィグレーション[HW interrupt:] ドロップダウンリストと ATTACH / DETACH パラメータの OB_NR ドロップ ダウンリストに表示されます。

EVENT パラメータ

ハードウェア割り込みイベントが有効になると、一意のデフォルトイベント名がこの特定のイベントに割り付けられます。 このイベント名は[Event name:]編集ボックスを編集して変更できますが、一意の名前にする必要があります。 これらのイベント名は [Constants]タグテーブルのタグ名になり、ATTACH および DETACH 命令ボックスの EVENT パラメータドロップダウンリストに表示されます。 タグの値は、イベントの識別に使用する内部番号です。

一般操作

各ハードウェアイベントは、ハードウェア割り込み OB にアタッチして、ハードウェア割り込みイベントの発生時にキュー登録することができます。 OB イベントのアタッチメントは、コンフィグレーション時またはランタイムで可能です。

コンフィグレーション時に有効化されたイベントに、OBをアタッチまたはデタッチすることができます。 コンフィグレーション時に OBをイベントにアタッチするには、[HW interrupt:] ドロップダウンリストを使用して(右側の下矢印をクリックします)、使用可能なハードウェア割り込み OBのリストから OBを選択します。 このリストから適切な OB名を選択するか、[<not connected>]を選択してアタッチメントを解除します。

有効化されたハードウェア割り込みイベントをランタイム中にアタッチまたはデタッチすることもできます。 ランタイム中に ATTACH または DETACH プログラム命令を(必要回数)使用して、有効化された割り込みイベントを適切な OB でアタッチまたはデタッチできます。 OB が現在アタッチされていない場合は(デバイスコンフィグレーションの[<not connected>]選択項目、または DETACH 命令実行の結果として)、有効化されたハードウェア割り込みイベントが無視されます。

DETACH 操作

DETACH 命令を使用して、特定のイベントまたはすべてのイベントを特定の OB から デタッチすることができます。 EVENT が指定されている場合は、この 1 つのイベント だけが指定された OB_NR からデタッチされ、この OB_NR に現在接続されているその 他すべてのイベントはアタッチされたままになります。 EVENT が指定されていない場合は、OB NR に現在アタッチされているすべてのイベントがデタッチされます。

条件コード

RET_VAL (W#16#)	ENO のス テータス	説明	
0000	1	エラーは発生していません。	
0001	0	デタッチするものがありません(DETACH のみ)	
8090	0	OB が存在しません	
8091	0	OB のタイプが不正です	
8093	0	イベントが存在しません	

6.2.6.2 時間遅延割り込み命令の開始およびキャンセル

SRT_DINT 命令と CAN_DINT 命令を使って、時間遅延割り込みの処理を開始およびキャンセルできます。 各時間遅延割り込みは、指定した遅延時間の後に発生する一度限りのイベントです。 時間遅延イベントが、時間遅延の期限が切れる前にキャンセルされた場合、プログラム割り込みは発生しません。





SRT_DINT は、パラメータ DTIME で指定された遅延時間が 経過すると、OB (オーガニゼーションブロック)サブプログ ラムを実行する時間遅延割り込みを開始します。

CAN_DINT は、既に開始された時間遅延割り込みをキャンセルします。 時間遅延割り込み OB は、この場合には実行されません。

SRT_DINT パラメータ

パラメー タ	パラメー タタイプ	データタ イプ	説明
OB_NR	IN	Int	時間遅延後に開始されるオーガニゼーションブロック(OB): プロジェクトツリー内の[Add new block] を使用して、利用可能な割り込み OB から選択します。 このパラメータフィールドをダブルクリックしてから、ヘルパーアイコンをクリックして利用可能な OB を表示します。
DTIME	IN	Time	時間遅延の値 (1~60000 ms) より長い遅延時間を作成するには、例えば時間遅 延割り込み OB 内のカウンタを使用します。
SIGN	IN	Word	S7-1200 では使用しません。任意の値を受け付けます
RET_VAL	OUT	Int	実行条件コード

CAN_DINT パラメータ

パラメー タ	パラメー タタイプ	データタ イプ	説明
OB_NR	IN	Int	時間遅延割り込み OB 識別子。 OB 番号またはシンボル名を使用できます。
RET_VAL	OUT	Int	実行条件コード

操作

SRT_DINT 命令は、時間遅延を指定し、内部時間遅延タイマを開始し、時間遅延割り込み OB サブプログラムを時間遅延タイムアウトイベントと関連付けます。 指定された時間遅延が経過すると、関連付けられた時間遅延 OB の実行をトリガするプログラム割り込みが生成されます。 CAN_DINT 命令を実行することによって、指定された時間遅延が発生する前に、プロセス中の時間遅延割り込みをキャンセルすることができます。有効な時間遅延と時間サイクリック割り込みイベントの合計数は、4 を超えてはなりません。

プロジェクトへの時間遅延割り込み OB の追加

SRT_DINT 命令と CAN_DINT 命令に割り付け可能なのは、時間遅延割り込み OB のみです。 時間遅延割り込み OB は新規プロジェクトに存在しません。 時間遅延割り込み OB をプロジェクトに追加する必要があります。 時間遅延割り込み OB を作成するには、次の手順に従います。

- 1. プロジェクトツリーの[Program blocks]分岐にある[Add new block]をダブルクリックして、[Organization block (OB)]、[Time delay interrupt]の順に選択します。
- 2. OB の名前変更、プログラミング言語の選択、またはブロック番号の選択のオプションがあります。 自動的に割り付けられた番号ではないブロック番号を割り付ける場合には、手動割り付けに切り替えます。
- 3. 時間遅延割り込み OB サブプログラムを編集し、時間遅延タイムアウトイベント発生時に実行させるプログラム応答を作成します。 時間遅延割り込み OB から他の FC プログラムブロックと FB プログラムブロックを呼び出すことができます。最大ネスト深さは 4 です。
- **4.** 新しく割り付けられた時間遅延割り込み **OB** 名は、**SRT_DINT** 命令と **CAN_DINT** 命令の **OB_NR** パラメータの編集時に利用可能になります。

条件コード

RET_VAL (W#16#)	説明
0000	エラーは発生していません。
8090	パラメータ OB_NR が不正です。
8091	パラメータ DTIME が不正です。
80A0	時間遅延割り込みが起動していません。

6.2.6.3 アラーム割り込み命令の無効化と有効化

DIS_AIRT 命令と EN_AIRT 命令を使用して、アラーム割り込み処理を無効または有効にします。



DIS_AIRT は新規割り込みイベントの処理を遅延させます。 DIS_AIRT を OB で複数回実行できます。 DIS_AIRT の実行は、オペレーティングシステムでカウントされます。 これらの呼び出しはそれぞれ、特に "EN_AIRT"命令により再びキャンセルされるまで、あるいは現在の OB が完全に処理されるまで実行されます。

DIS_AIRT が再び有効になると、DIS_AIRT が有効な間に発生した割り込みは処理され、また現在の OB が実行されるとただちに割り込みが処理されます。



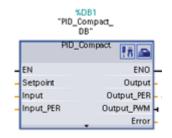
EN_AIRT は、以前 DIS_AIRT 命令を使って無効にした割り込みイベントの処理を有効にすることができます。 各 DIS_AIRT の実行が、EN_AIRT の実行によってキャンセルされる必要があります。 例えば、DIS_AIRT を 5 回実行して割り込みを無効にした場合、EN_AIRT を 5 回実行してこれをキャンセルする必要があります。

EN_AIRT は、特定の OB について割り込みが再び有効になる前に、同じ OB 内、または同じ OB から呼び出された任意の FC または FB で実行する必要があります。

パラメータ RET_VAL は、割り込み処理が無効にされた回数(キューに登録された DIS_AIRT の実行数)を示します。 割り込み処理は、パラメータ RET_VAL = 0 になったときにのみ再び有効になります。

パラメータ	パラメータ タイプ	データタ イプ	説明
RET_VAL	OUT	INT	遅延数 = キューに登録された DIS_AIRT の実 行数

6.2.7 PIDコントロール



「PID_Compact」ステートメントによって、PID コントローラは、自動モードと手動モードで自己調整の最適化が可能になります。

PID_Compact 命令の詳細については、TIA ポータルのオンラインヘルプを参照してください。

6.2.8 モーションコントロール命令

モーションコントロール命令は、CPU の関連付けられたテクノロジーデータブロック と専用 PTO (パルストレイン出力)を使用して、軸の動作を制御します。 モーションコントロール命令の詳細については、STEP 7 Basic のオンラインヘルプを参照してください。

通知

パルス出力ジェネレータのパルス周波数は、CPU のデジタル出力で 100 KHz であり、シグナルボードのデジタル出力で 20 KHz です。 ただし、STEP 7 Basic は、最大速度またはこのハードウェア制限を超える周波数で軸を設定しても、警告を出しません。 これにより、アプリケーションで問題が発生する可能性があります。したがって、常にハードウェアの最大パルス周波数を超えないようにしてください。



MC_Power はモーションコン MC_Reset はすべてのモトロール軸を有効または無効 ーションコントロールエにします。 ラーをリセットします。



MC_Reset はすべてのモーションコントロールエラーをリセットします。 認識可能なすべてのモーションコントロールエラーの認識ができます。



MC_Home は、軸制御プログ ラムと軸機械位置決定システ ムとの関係を確立します。



MC_Halt は、すべての動 作プロセスをキャンセル し、軸動作を停止しま す。停止位置は定義され ません。



MC_MoveJog は、テスト とスタートアップでジョグ モードを実行します。



MC_MoveAbsolute は、絶対 位置制御の動作を開始しま す。 ジョブはターゲット位置 決め動作を開始します。 に達すると終了します。



MC_MoveRelative は、開 始位置に対する相対位置



MC_MoveVelocity によ り、軸は指定された速度で 移動します。

注記

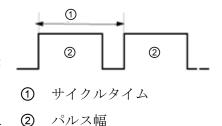
パルストレイン出力は、ユーザープログラムで他の命令によって使用することはでき ません。

CPU またはシグナルボードの出力をパルスジェネレータとして設定する(PWM または 基本モーションコントロール命令と共に使用する)場合、対応する出力アドレス(Q0.0、 Q0.1、Q4.0、Q4.1)はQメモリから削除され、ユーザープログラムの他の目的に使用す ることはできません。 ユーザープログラムがパルスジェネレータとして使用されてい る出力に値を書き込むと、その CPU はこの値を物理出力に書き込みません。

6.2.9 パルス命令

6.2.9.1 CTRL_PWM命令

CTRL_PWM Pulse Width Modulation (PWM)命令では、可変デューティサイクルによる固定サイクルタイム出力が可能です。 PWM 出力は、開始後に指定された周波数(サイクルタイム)で継続的に実行されます。



パルス幅は、目的の制御を行えるように必要に応じて変更可能です。

パルス幅はサイクルタイムの 100 分の 1(0~100)、1000 分の 1(0~1000)、10,000 分の 1(0~10000)、または S7 アナログフォーマットとして表現できます。 パルス幅は S7 の (パルスなし、常にオフ)からフルスケール(パルスなし、常にオン)まで変更可能です。

PWM 出力は 0 からフルスケールまで可変であるため、アナログ出力と似たデジタル出力を提供します。 例えば、PWM 出力はモーターの速度を停止からフルスピードまで制御するために使用できます。または、バルブの位置を閉じた状態から完全に開いた状態まで制御するために使用できます。

2つのパルスジェネレータが、高速パルス出力ファンクションの制御に利用できます。 PWM およびパルストレイン出力(PTO)。 PTO はモーションコントロール命令で使用されます。 各パルスジェネレータを PWM または PTO のいずれかに割り付けることはできますが、同時に両方に割り付けることはできません。

2つのパルスジェネレータが、以下の表に示すように特定のデジタル出力にマッピングされています。 オンボード CPU 出力、またはオプションのシグナルボード出力を使用できます。 出力点番号を下の表に示します(デフォルト出力コンフィグレーションを想定)。 デフォルトの出力点を変更した場合、出力点の番号は割り付けたものになります。このことには関係なく、CPU または接続されたシグナルボードで、PTO1/PWM1 は最初の 2 つのデジタル出力を使用し、PTO2/PWM2 は次の 2 つの出力を使用します。PWM には 1 つの出力だけが必要ですが、PTO はチャンネル当たりオプションで 2 つの出力を使用することができます。 出力がパルスファンクションに必要ない場合、他の用途に利用できます。

説明	デフォルトの出力割り付け		
		パルス	方向
PTO 1	オンボード CPU	Q0.0	Q0.1
	シグナルボード	Q4.0	Q4.1
PWM 1	オンボード CPU	Q0.0	
	シグナルボード	Q4.0	
PTO 2	オンボード CPU	Q0.2	Q0.3
	シグナルボード	Q4.2	Q4.3
PWM 2	オンボード CPU	Q0.2	
	シグナルボード	Q4.2	

PWM のパルスチャンネルのコンフィグレーション

PWM 操作を準備するには、最初に CPU、パルスジェネレータ(PTO/PWM)、PWM1 または PWM2 を選択し、デバイスコンフィグレーションでパルスチャンネルを設定します。 パルスジェネレータ(チェックボックス)を有効にします。 パルスジェネレータを有効にすると、一意のデフォルト名がこの特定のパルスジェネレータに割り付けられます。 この名前は[Name:]編集ボックスを編集して変更できますが、一意の名前にする必要があります。 有効にされたパルスジェネレータは[constant]タグテーブルのタグになり、CTRL_PWM 命令の PWM パラメータとして使用できるようになります。

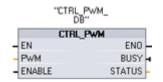
通知

パルス出力ジェネレータのパルス周波数は、CPU のデジタル出力で 100 KHz であり、シグナルボードのデジタル出力で 20 KHz です。 ただし、STEP 7 Basic は、最大速度またはこのハードウェア制限を超える周波数で軸を設定しても、警告を出しません。 これにより、アプリケーションで問題が発生する可能性があります。したがって、常にハードウェアの最大パルス周波数を超えないようにしてください。

パルスジェネレータの名前変更、コメントの追加をすることができ、以下のようにパラメータを割り付けます。

- パルスジェネレータを次のように使用します。 PWM または PTO (PWM を選択)
- 出力ソース: オンボード CPU またはシグナルボード
- タイムベース: ミリ秒またはマイクロ秒
- パルス幅フォーマット:
 - 100分の1(0~100)
 - 1000分の1(0~1000)
 - 10,000 分の 1(0~10000)
 - S7 アナログフォーマット(0~27648)
- Cycle time (サイクルタイム): サイクルタイム値を入力します。 この値はデバイスコンフィグレーションでのみ変更できます。
- 初期パルス幅: 初期パルス幅の値を入力します。 ランタイム中にパルス幅の値を変 更することができます。

出力アドレス



開始アドレス: パルス幅の値を持つ Q ワードアドレスを入力します。 デフォルトの場所は PWM1 で QW1000、PWM2 で QW1002 です。 この場所の値はパルス幅を制御し、上記で指定された[Initial pulse width:]値に初期化されます。 初期化は、CPU が STOP から RUN モードに移行するたびに行われます。 この Q ワード値をランタイム中に変更すると、パルス幅が変更されます。

パラメータ		データ タイプ	初期値	説明
PWM	IN	Word	0	PWM 識別子: 有効にされたパルスジェネレータの名 前は[constant]タグテーブルのタグに なり、PWM パラメータとして使用で きるようになります。
ENABLE	IN	Bool		1=開始パルスジェネレータ 0 =停止パルスジェネレータ
BUSY	OUT	Bool	0	ファンクションビジー
STATUS	OUT	Word	0	実行条件コード

操作

データブロック(DB)は、パラメータ情報を保存するため CTRL_PWM 命令で使用されます。 CTRL_PWM 命令をプログラムエディタに書き込むと、DB が割り付けられます。 データブロックパラメータはユーザーが個別に変更できず、CTRL_PWM 命令によって制御されます。

PWM パラメータのタグ名を使用して、使用する有効にされたパルスジェネレータを指定します。

EN 入力が TRUE の場合、PWM_CTRL 命令は、ENABLE 入力での値に基づいて、識別 された PWM を開始または停止します。 パルス幅は、Q ワード出力アドレスに関連付けられた値により指定されます。

S7-1200 は、CTRL_PWM 命令の実行時に要求を処理するので、パラメータ BUSY は **S7-1200** CPU モデルで常に FALSE をレポートします。

エラーが検出されると、ENO は FALSE にセットされ、パラメータ STATUS には条件 コードが入ります。

パルス幅は、PLC が最初に RUN モードに入ると、デバイスコンフィグレーションで設定された初期値に設定されます。 パルス幅を変更する必要があれば、値をデバイスコンフィグレーションで指定された Q ワードの場所に書き込みます ([出力アドレス] / [開始アドレス:])。 ムーブ、変換、演算、または PID ボックスなどの命令を使用して、必要なパルス幅を適切な Q ワードに書き込みます。 Q ワード値の有効な範囲を使用してください(パーセント、1000 分の 1、10,000 分の 1、または S7 アナログフォーマット)。

条件コード

STATUS 値	説明
0	エラーは発生していません。
80A1	PWM 識別子が有効な PWM をアドレス指定していません。

PWM、PTO に割り付けられたデジタル I/O ポイントは強制できません

パルス幅変調(PWM)、パルストレイン出力(PTO)デバイスで使用されるデジタル I/O ポイントは、デバイスコンフィグレーション内で割り付けられます。 これらのデバイス にデジタル I/O ポイントのアドレスが割り付けられると、割り付けられた I/O ポイントのアドレスをモニタテーブルの強制ファンクションで変更することはできません。

6.3 グローバルライブラリ命令

パルストレイン出力は、ユーザープログラムで他の命令によって使用することはできません。

CPU またはシグナルボードの出力をパルスジェネレータとして設定する(PWM または基本モーションコントロール命令と共に使用する)場合、対応する出力アドレス(Q0.0、Q0.1、Q4.0、Q4.1)はQメモリから削除され、ユーザープログラムの他の目的に使用することはできません。 ユーザープログラムがパルスジェネレータとして使用されている出力に値を書き込むと、その CPU はこの値を物理出力に書き込みません。

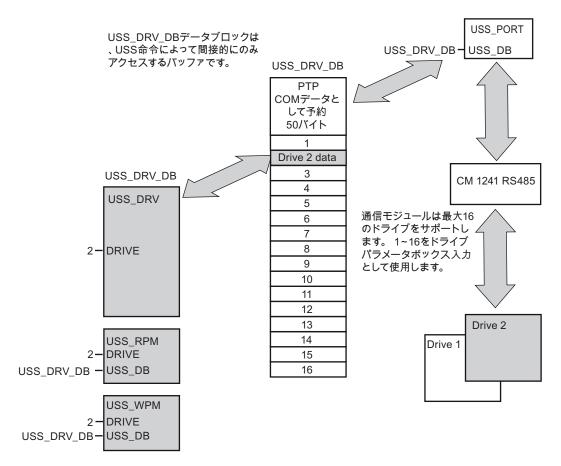
6.3 グローバルライブラリ命令

6.3.1 USS

USS プロトコルライブラリは、USS プロトコルをサポートするシーメンスのドライブを制御します。 命令には、ドライブと通信を行うため USS プロトコルを使用するように設計されたファンクションが含まれます。 CM 1241 RS485 モジュールは、RS485 ポートのドライブと通信を行います。 USS ライブラリを使って、物理ドライブや読み書きドライブパラメータを制御できます。

6.3.1.1 USSプロトコルの使用要件

ライブラリは、USS プロトコルをサポートするため 1 FB および 3 FC を提供しています。 各 CM 1241 RS485 通信モジュールは、最大 16 のドライブをサポートします。 シングルインスタンスのデータブロックには、インストールする各ポイントツーポイント通信モジュールに接続された、USS ネットワーク上のすべてのドライブに関する一時的なストレージとバッファが含まれます。 これらのドライブの USS ファンクションは、このデータブロックの情報を共有します。



1 台の CM 1241 RS485 に接続されたすべてのドライブ(最大 16)は、同じ USS ネットワークの一部です。 別の CM 1241 RS485 に接続されたすべてのドライブは、別の USS ネットワークの一部です。 S7-1200 は最大 3 台の CM 1241 RS485 デバイスをサポートするため、最大 3 つの USS ネットワークを持つことができます。各ネットワークには最大 16 のドライブ、合計で 48 の USS ドライブがサポートされます。

各 USS ネットワークは一意のデータブロックを使用して管理されます(3 つの CM 1241 RS485 デバイスを使用する 3 つの USS ネットワークには、3 つのデータブロックが必要です)。 1 つの USS ネットワークに関連付けられたすべての命令は、このデータブロックを共有する必要があります。 これにはすべての USS_DRV 命令、USS_PORT 命令、

6.3 グローバルライブラリ命令

USS_RPM 命令、USS_WPM 命令が含まれ、1 つの USS ネットワーク上でのすべての ドライブ制御に使用されています。

USS_DRV 命令はファンクションブロック(FB)です。 USS_DRV 命令をエディタに書き込むと、この FB についてどの DB を割り付けるかを指定するための[Call options]ダイアログが表示されます。 これが、この USS ネットワークのプログラムでの最初のUSS_DRV 命令であれば、デフォルトの DB 割り付けを受け入れると(または必要に応じて名前を変更し)、新規 DB が作成されます。 ただし、このチャンネルの最初のUSS_DRV 命令ではない場合は、[Call options]ダイアログ内のドロップダウンリストを使って、以前にこの USS ネットワークに割り付けられた適切な DB を使用する必要があります。

命令 USS_PORT、USS_RPM、USS_WPM はすべてファンクションです(FC)。 これらの FC をエディタに書き込んでも、DB は割り付けられません。 そのかわりに、適切な DB をこれらの命令の「USS_DB」入力に割り付けてください(このパラメータフィールドをダブルクリックしてから、使用可能な DB を表示するヘルパーアイコンをクリックします)。

USS_PORT ポートファンクションは、CPU とポイントツーポイント通信モジュール経由のドライブ間との実際の通信を処理します。 このファンクションを呼び出すたびに、1台のドライブで1つの通信が処理されます。 ユーザープログラムは、このファンクションを十分な速度で呼び出して、ドライブによる通信タイムアウトを防止する必要があります。 このファンクションは、メインまたは割り込み OB で呼び出すことができます。

USS_DRV ファンクションブロックは、プログラムから USS ネットワーク上の指定されたドライブへのアクセスを可能にします。 その入力と出力は、ドライブのステータスと制御です。 ネットワーク上に 16 台のドライブがある場合、ユーザープログラムは、各ドライブについて少なくとも 16 の USS_DRV 呼び出しが必要です。 これらのブロックは、ドライブのファンクションを制御するために必要な速度で呼び出される必要があります。

USS DRV ファンクションブロックをメイン OB からのみ呼び出すことができます。

注意

メイン OB からは USS_DRV 、 USS_RPM 、 USS_WPM だけを呼び出します。

USS_PORT ファンクションは任意の OB から呼び出しが可能で、通常は時間遅延割り込みから呼び出されます。

USS_PORT の割り込み防止に失敗すると、予期しないエラーが発生する可能性があります。

USS_RPM ファンクションと USS_WPM ファンクションは、リモートドライブ操作パラメータを読み書きします。 これらのパラメータは、ドライブの内部操作を制御します。 これらのパラメータの定義については、ドライブマニュアルを参照してください。ユーザープログラムには必要な数のファンクションを含めることができますが、ドライブ当たり、一度に有効になる読み取りまたは書き込み要求は 1 つだけです。 USS_RPMファンクションと USS_WPM ファンクションは、メイン OB からのみ呼び出すことができます。

ドライブとの通信に必要な時間の計算

ドライブとの通信は、**S7-1200** スキャンに対して非同期です。 通常、**S7-1200** は、**1** つのドライブ通信トランザクションが完了する前に複数のスキャンを完了します。

USS_PORT 間隔は、1 つのドライブトランザクションに必要な時間です。 以下の表は、各ボーレートに対する最小の USS_PORT 間隔を示しています。 USS_PORT ファンクションを USS_PORT 間隔よりも高い頻度で呼び出しても、トランザクションの数は増えません。 ドライブタイムアウト間隔は、通信エラーによりトランザクション完了のため 3 回の試行が行われた場合、あるトランザクションにかかったと思われる時間の量です。 デフォルトで、USS プロトコルライブラリは各トランザクションで自動的に最大 2 回の試行を行います。

ボーレート	計算された最小 USS_PORT 呼び出 し 間隔(ミリ秒)	ドライブ当たりのドライブメッセージ間隔タイムアウト(ミリ 秒)
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

6.3.1.2 USS_DRV命令

USS_DRV 命令は、要求メッセージを作成し、ドライブ応答メッセージを解釈することで、ドライブとデータを交換します。 各ドライブについて個別のファンクションブロックを使用する必要がありますが、1 つの USS ネットワークおよびポイントツーポイント通信モジュールに関連付けられたすべての USS ファンクションは、同じインスタンスのデータブロックを使用しなければなりません。 最初の USS_DRV 命令を書き込むときに DB 名を作成し、最初に作成したその DB を再使用する必要があります。

最初に USS_DRV を実行すると、USS アドレスで示されたドライブ(パラメータ DRIVE)はインスタンス DB により初期化されます。 この初期化の後、USS_PORT を引き続き実行すると、このドライブ番号のドライブとの通信を開始できます。

ドライブ番号を変更するには、インスタンス DB を初期化する PLC の STOP から RUN モードへの移行が必要です。 入力パラメータは USS TX メッセージバッファ内に設定され、出力は、存在する場合、「以前の」有効な応答バッファから読み取られます。 USS_DRV の実行中にデータ転送は発生しません。 ドライブは、USS_PORT が実行されると通信を行います。 USS_DRV は、送信されるメッセージだけを設定し、以前の要求で受信された可能性があるデータを解釈します。

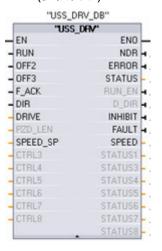
ドライブの回転方向は、DIR 入力(BOOL)を使用して、または符号(正または負)を SPEED_SP 入力(REAL)と共に使用して制御できます。 以下の表は、これらの入力がどのように組み合わされて動作し、ドライブ方向を決めるかを示しています(モーターが前方への回転で配線されていると想定しています)。

SPEED_SP	DIR	ドライブ方向
值 > 0	0	後方
値 > 0	1	前方
值 < 0	0	前方
値 < 0	1	後方

LAD (デフォルト表示)



LAD (拡張表示)



ボックス下部をクリック すると、ボックスが拡張 され、すべてのパラメー タが表示されます。

グレー表示のパラメータ はオプションで、割り付 けは必須ではありませ ん。

パラメータ		データタ イプ	説明
RUN	IN	Bool	ドライブ開始ビット: True の場合、入力によってドライブは事前設定された速度で動作します。
OFF2	IN	Bool	電気ストップビット: False の場合、このビットによってドライブは制動なしで惰行運転して停止します。
OFF3	IN	Bool	高速停止ビット - False の場合、ドライブを惰行 運転して停止させるのではなく、ドライブを制 動することで高速停止が発生します。
F_ACK	IN	Bool	故障認識ビット-このビットは、ドライブ上の 故障ビットをリセットするためセットされま す。 このビットは故障が修正された後にセット され、以前の故障を示す必要がなくなったこと を表します。
DIR	IN	Bool	ドライブ方向制御 - このビットは、方向が前方 であることを示すためセットされます(正の SPEED_SP の場合)。

プログラミング手順 6.3 グローバルライブラリ命令

パラメータ		データタ イプ	説明
DRIVE	IN	USInt	ドライブアドレス: この入力は USS ドライブの アドレスです。 有効な範囲はドライブ 1~ドラ イブ 16 です。
PZD_LEN	IN	USInt	ワード長 - これは PZD データのワード数です。 有効な値は 2、4、6、または 8 ワードです。 デ フォルトは 2 です。
SPEED_S P	IN	Real	速度セットポイント - 設定済み周波数のパーセンテージとしてのドライブ速度です。 正の値は前方方向を示します(DIR が True の場合)。
CTRL3	IN	UInt	コントロールワード 3 - ドライブのユーザー設定可能パラメータに書き込まれる値。 この値はユーザーがドライブに設定する必要があります。 オプションのパラメータです。
CTRL4	IN	UInt	コントロールワード 4 - ドライブのユーザー設定可能パラメータに書き込まれる値。 この値はユーザーがドライブに設定する必要があります。 オプションのパラメータです。
CTRL5	IN	UInt	コントロールワード 5 - ドライブのユーザー設定可能パラメータに書き込まれる値。 この値はユーザーがドライブに設定する必要があります。 オプションのパラメータです。
CTRL6	IN	UInt	コントロールワード 6 - ドライブのユーザー設定可能パラメータに書き込まれる値。この値はユーザーがドライブに設定する必要があります。
CTRL7	IN	UInt	コントロールワード 7 - ドライブのユーザー設定可能パラメータに書き込まれる値。 この値はユーザーがドライブに設定する必要があります。 オプションのパラメータです。
CTRL8	IN	UInt	コントロールワード 8 - ドライブのユーザー設定可能パラメータに書き込まれる値。 この値はユーザーがドライブに設定する必要があります。 オプションのパラメータです。

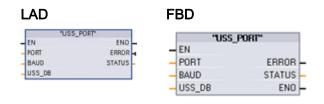
プログラミング手順 6.3 グローバルライブラリ命令

パラメータ	パラメー タタイプ	データタ イプ	説明
NDR	OUT	Bool	新規データ有効 - このビットが True の場合、出力に新規通信要求からのデータが含まれていることを示します。
ERROR	ОИТ	Bool	エラーが発生 - True の場合、エラーが発生し、 STATUS 出力が有効であることを示します。 そ の他すべての出力は、エラーでゼロにセットさ れます。 通信エラーは、USS_PORT 命令の ERROR 出力と STATUS 出力でのみレポートさ れます。
STATUS	OUT	UInt	要求のステータス値。 スキャンの結果を示します。 これはドライブから返されるステータスワードではありません。
RUN_EN	OUT	Bool	実行の有効化 - このビットはドライブが運転中であるかどうかを示します。
D_DIR	OUT	Bool	ドライブ方向 - このビットはドライブが前方に 運転中であるかどうかを示します。
INHIBIT	OUT	Bool	ドライブ禁止 - このビットは、ドライブ上の禁 止ビットの状態を示します。
FAULT	ОИТ	Bool	ドライブ故障 - このビットは、ドライブが故障 を登録したことを示します。 ユーザーはこの問 題を修正してから、F_ACK ビットをセットし て、セットされたこのビットをクリアする必要 があります。
SPEED	OUT	REAL	ドライブの現在速度(ドライブステータスワード 2 のスケール値) - 設定速度のパーセンテージとしてのドライブ速度値。
STATUS1	OUT	UInt	ドライブステータスワード 1 - この値にはドラ イブの固定ステータスビットが含まれます。
STATUS3	OUT	UInt	ドライブステータスワード 3 - この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。

パラメータ	パラメー タタイプ	データタ イプ	説明
STATUS4	OUT	UInt	ドライブステータスワード 4 - この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。
STATUS5	OUT	UInt	ドライブステータスワード 5 - この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。
STATUS6	OUT	UInt	ドライブステータスワード 6 - この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。
STATUS7	OUT	UInt	ドライブステータスワード 7 -この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。
STATUS8	OUT	UInt	ドライブステータスワード 8 - この値には、ドライブ上のユーザー設定可能ステータスワードが含まれます。

6.3.1.3 USS_PORT命令

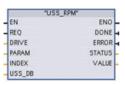
USS_PORT 命令は、USS ネットワーク経由で通信を処理します。 通常、プログラム内のポイントツーポイント通信モジュール当たり 1 つだけの USS_PORT ファンクションが存在し、このファンクションの各呼び出しが、1 つのドライブの送受信を処理します。 ユーザープログラムは、USS_PORT ファンクションをドライブタイムアウトがないように十分な頻度で実行する必要があります。 USS ネットワークおよびポイントツーポイント通信モジュールに関連付けられたすべての USS ファンクションは、同じインスタンスデータブロックを使用する必要があります。 通常、USS_PORT は時間遅延割り込み OB から呼び出され、ドライブタイムアウトを防止し、USS_DRV 呼び出しで利用可能な最新の USS データを保管します。



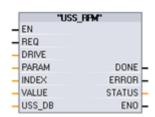
パラメー タ	パラメー タタイプ	データタ イプ	説明
PORT	IN	Port	ポイントツーポイント通信モジュール。 識別子: デフォルトタグテーブルの[Constants]タブで参照 できる定数です。
BAUD	IN	DInt	USS 通信に使用されるボーレート。
USS_DB	IN	DInt	USS_DRV 命令がユーザープログラムに書き込まれるとき作成および初期化される、インスタンスDBへの参照。
ERROR	OUT	Bool	True の場合、このピンは、エラーが発生し、 STATUS 出力が有効であることを示します。
STATUS	OUT	UInt	要求のステータス値。 スキャンまたは初期化の結果を示します。 追加情報は、一部のステータスコードに関して"USS_Extended_Error"変数に記載されています。

6.3.1.4 USS_RPM命令

LAD



FBD



USS_RPM 命令は、ドライブから パラメータを読み取ります。 USS ネットワークおよびポイントツー ポイント通信モジュールに関連付 けられたすべての USS ファンクションは、同じデータブロックを使 用する必要があります。

USS_RPM はメイン OB から呼び 出す必要があります。

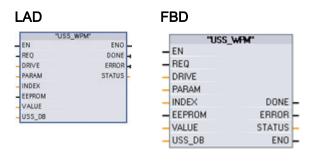
プログラミング手順 6.3 グローバルライブラリ命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
REQ	IN	Bool	送信要求: True の場合、新規読み取り要求があることを示します。 このパラメータの要求が既に保留中の場合は無視されます。
DRIVE	IN	USInt	ドライブアドレス: この入力は USS ドライブのアドレスです。 有効な範囲はドライブ 1~ドライブ16 です。
PARAM	IN	UInt	パラメータ番号: この入力は、書き込まれるドライブパラメータを示します。 このパラメータの範囲は 0~2047です。この範囲を超えるパラメータにアクセスする方法の詳細については、ドライブのマニュアルを参照してください。
INDEX	IN	UInt	パラメータインデックス: この入力は、書き込まれるドライブパラメータインデックスを示します。 16 ビット値。最下位バイトは、0~255 の範囲の実際のインデックス値です。 最上位バイトもドライブで使用することができ、ドライブ固有です。詳細は、ドライブのマニュアルを参照してください。
USS_DB	IN	Variant	USS_DRV 命令がユーザープログラムに書き込まれるとき作成および初期化される、インスタンスDBへの参照。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	読み取られたパラメータの値で、DONE ビットが True の場合のみ有効です。

プログラミング手順 6.3 グローバルライブラリ命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
DONE	OUT	Bool	完了: TRUE の場合、VALUE 出力が以前に要求された読み取りパラメータ値を保持していることを示します。
			このビットは、USS_DRVがドライブからの読み 取り応答を確認するとセットされます。
			このビットは以下のいずれかの場合にリセットされます。
			• 他の USS_RPM ポーリングにより応答データを 要求する
			または
			• USS_DRV への次の 2 つの呼び出しの 2 番目で
ERROR	OUT	Bool	エラーが発生 - True の場合、エラーが発生し、 STATUS 出力が有効であることを示します。 その 他すべての出力は、エラーでゼロにセットされま す。 通信エラーは、USS_PORT 命令の ERROR 出力と STATUS 出力でのみレポートされます。
STATUS	OUT	UInt	要求のステータス値です。 読み取り要求の結果を示します。 追加情報は、一部のステータスコードに関して"USS_Extended_Error"変数に記載されています。

6.3.1.5 USS_WPM命令



USS_WPM 命令は、ドライブのパラメータを修正します。 USS ネットワークおよびポイントツーポイント通信モジュールに関連付けられたすべての USS ファンクションは、同じデータブロックを使用する必要があります。 USS_WPM はメイン OB から呼び出す必要があります。

注記

EEPROM 書き込み操作

EEPROM の永続書き込み操作の過度の使用に注意してください。 EEPROM の寿命を 延ばすため、EEPROM 書き込み操作回数は最小限にしてください。

パラメー タ	パラメー タタイプ	データタ イプ	説明
REQ	IN	Bool	送信要求: True の場合、新規書き込み要求がある ことを示します。 このパラメータの要求が既に保 留中の場合は無視されます。
DRIVE	IN	USInt	ドライブアドレス: この入力は USS ドライブのアドレスです。 有効な範囲はドライブ 1~ドライブ16 です。
PARAM	IN	UInt	パラメータ番号: この入力は、書き込まれるドライブパラメータを示します。 このパラメータの範囲は 0~2047です。この範囲を超えるパラメータにアクセスする方法の詳細については、ドライブのマニュアルを参照してください。

プログラミング手順 6.3 グローバルライブラリ命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
INDEX	IN	UInt	パラメータインデックス: この入力は、書き込まれるドライブパラメータインデックスを示します。 16 ビット値。最下位バイトは、0~255 の範囲の実際のインデックス値です。 最上位バイトもドライブで使用することができ、ドライブ固有です。詳細は、ドライブのマニュアルを参照してください。
EEPROM	IN	Bool	ドライブの EEPROM に保存: True の場合、ドライブパラメータへの書き込みはドライブの EEPROM に保存されます。 False の場合、書き込みは一時的であり、ドライブのパワーサイクルが行われると保持されません。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	書き込まれるパラメータの値。 REQ の移行で有効でなければなりません。
USS_DB	IN	Variant	USS_DRV 命令がユーザープログラムに書き込まれるとき作成および初期化される、インスタンスDBへの参照。
DONE	OUT	Bool	完了: TRUE は、入力 VALUE がドライブに書き込まれたことを示します。 このビットは、USS_DRV がドライブからの書き込み応答を確認するとセットされます。 このビットは以下のいずれかの場合にリセットされます。 別の USS_WPM ポーリングにより、またはUSS_DRV への次の 2 つの呼び出しの 2 番目で書き込みが完了したというドライブの確認を要求する場合。

パラメー タ	パラメー タタイプ	データタ イプ	説明
ERROR	OUT	Bool	エラーが発生しました。 True の場合、エラーが発生し、STATUS 出力が有効であることを示します。 その他すべての出力は、エラーでゼロにセットされます。 通信エラーは、USS_PORT 命令のERROR 出力と STATUS 出力でのみレポートされます。
STATUS	OUT	UInt	要求のステータス値です。 書き込み要求の結果を示します。 追加情報は、一部のステータスコードに関して"USS_Extended_Error"変数に記載されています。

6.3.1.6 USSステータスコード

USS 命令のステータスコードは、USS ファンクションの STATUS 出力で返されます。

STATUS 値 (W#16#)	説明
0000	エラーは発生していません。
8180	ドライブ応答の長さが、ドライブから受信した文字に一致しませんでした。 エラーが発生したドライブ番号は、"USS_Extended_Error"変数によって返されます。 この表の下にある拡張エラーの説明を参照してください。
8181	VALUE パラメータが Word、Real、または DWord データタイプではありませんでした。
8182	パラメータ値に Word が入力され、ドライブ応答として DWord または Real を受信しました。
8183	パラメータ値に DWord または Real が入力され、ドライブ応答として Word を受信しました。
8184	ドライブからの応答テレグラムに不良なチェックサムがありました。 エラーが発生した ドライブ番号は、"USS_Extended_Error"変数によって返されます。 この表の下にある拡 張エラーの説明を参照してください。
8185	不正なドライブアドレスです(有効なドライブアドレス範囲: 1-16)
8186	速度設定点が有効範囲外です(有効な速度設定点の範囲: -200%~200%)

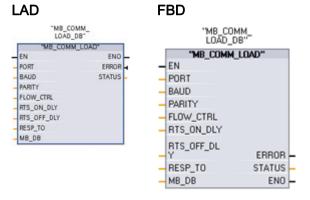
STATUS 値 (W#16#)	説明
8187	不正なドライブ番号が、送信された要求に応答しました。 エラーが発生したドライブ番号は、"USS_Extended_Error"変数によって返されます。 この表の下にある拡張エラーの説明を参照してください。
8188	不正な PZD ワード長が指定されました(有効な範囲 = 2、4、6 または 8 ワード)
8189	不正なボーレートが指定されました
818A	パラメータ要求チャンネルは、このドライブの別の要求によって使用中です
818B	ドライブが要求と再試行に応答しませんでした。 エラーが発生したドライブ番号は、 "USS_Extended_Error"変数によって返されます。 この表の下にある拡張エラーの説明を 参照してください。
818C	ドライブが、パラメータ要求操作に関する拡張エラーを返しました。 この表の下にある 拡張エラーの説明を参照してください。
818D	ドライブが、パラメータ要求操作に関する不正アクセスエラーを返しました。 パラメータアクセス制限の可能性について、ドライブのマニュアルを参照してください
818E	ドライブが初期化されていません。 このエラーコードは、該当ドライブの USS_DRV が 1回も呼び出されていない場合に、USS_RPM または USS_WPM に返されます。 これにより、USS_DRV の最初のスキャンの初期化は、上書きによってドライブが新規エントリとして初期化されるため、保留パラメータ読み取りまたは書き込み要求の上書きがされません。 このエラーを修正するには、このドライブ番号の USS_DRV を呼び出します。
80Ax-80Fx	USS ライブラリにより呼び出されたポイントツーポイント通信 FB が、固有のエラーを返しました。 このエラーコード値は USS ライブラリにより修正されず、ポイントツーポイント命令記述で定義されます。

USS ドライブの拡張エラーコード

USS ドライブは、ドライブの内部パラメータへの読み書きアクセスをサポートしています。この機能によって、ドライブのリモート制御とコンフィグレーションが可能です。ドライブパラメータアクセス操作は、範囲外の値やドライブの現在のモードに関する不正な要求などのエラーにより失敗する可能性があります。ドライブは、USS_DRV インスタンス DB の"USS_Extended_Error"変数で返されるエラーコード値を生成します。このエラーコード値は、USS_RPM命令またはUSS_WPM命令の最後の実行についてのみ有効です。ドライブエラーコードは、STATUSコードの値が16進数818Cである場合に、"USS_Extended_Error"変数に入ります。"USS_Extended_Error"のエラーコード値は、ドライブモデルによって変わります。パラメータの読み書き操作の拡張エラーコードの説明については、ドライブのマニュアルを参照してください。

6.3.2 MODBUS

6.3.2.1 MB_COMM_LOAD



MB_COMM_LOAD 命令は、ポイン トツーポイント(PtP) CM 1241 RS485 または Modbus RTU プロト コル通信の CM 1241 RS232 モジュ ールでのポートを設定します。

パラメータ	パラメータ タイプ	データタ イプ	説明
PORT	IN	UInt	通信ポート識別子:
			デバイスコンフィグレーションで CM モジュールをインストー
			ルすると、PORTボックス接続で利用可能なヘルパードロップ
			リストにポート識別子が表示されます。 この定数は、デフォ
54115			ルトタグテーブルの[Constants]タブでも参照できます。
BAUD	IN	UDInt	ボーレートの選択:
			300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600,
			76800, 115200
			他のすべての値は無効です
PARITY	IN	UInt	パリティ選択:
			• 0-なし
			• 1 – 奇数
			• 2 – 偶数
FLOW_CTRL	IN	UInt	フロー制御選択:
			• 0-(デフォルト)フロー制御なし
			• 1 – RTS が常に ON のハードウェアフロー制御(RS485 ポー
			トには適用されません)
			• 2 - RTS に切り替えられたハードウェアフロー制御

パラメータ	パラメータ	データタ	説明
	タイプ	イプ	PAG 2.1
RTS_ON_DL Y	IN	UInt	 RTS ON 遅延選択: 0 - (デフォルト) メッセージの最初の文字が送信されるまで RTS 有効の遅延なし 1~65535 - メッセージの最初の文字が送信されるまで RTS 有効の遅延(単位: ミリ秒)あり(RS-485 ポートには適用されません)。 RTS 遅延は FLOW_CTRL 選択とは関係なく適用されます。
RTS_OFF_DL Y	IN	UInt	 RTS OFF 遅延選択: 0 - (デフォルト) 最後の文字が送信されてから RTS が無効になるまで遅延なし 1~65535 - 最後の文字が送信されてから RTS が無効になるまで遅延(単位: ミリ秒)あり(RS-485 ポートには適用されません)。 RTS 遅延は FLOW_CTRL 選択とは関係なく適用されます。
RESP_TO	IN	UInt	応答タイムアウト: スレーブが応答するまで MB_MASTER が許可するミリ秒単位 の時間。 スレーブがこの時間内に応答しない場合、指定され た再試行数を送信すると、MB_MASTER は要求を再試行する か、要求をエラーで終了します。 5 ms~65535 ms (デフォルト値 = 1000ms)。
MB_DB	IN	Variant	MB_MASTER 命令または MB_SLAVE 命令によって使用されるインスタンスデータブロックへの参照。 MB_SLAVE または MB_MASTER をプログラムに書き込むと、MB_DB ボックス接続で利用可能なヘルパードロップリストに DB 識別子が表示されます。
ERROR	OUT	Bool	エラー: • 0 - エラーは検出されませんでした • 1 - エラーが検出され、パラメータ STATUS のエラーコードは有効です
STATUS	OUT	Word	ポートコンフィグレーションエラーコード

MB_COMM_LOAD は、Modbus RTU プロトコルのポートを設定するために実行されます。 ポートが設定されると、MB_SLAVE 命令または MB_MASTER 命令を実行して Modbus で通信を行います。

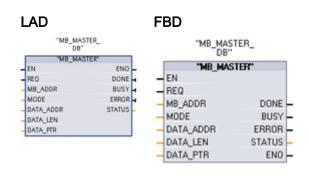
MB_COMM_LOAD は、ポートを初期化するため一度呼び出す必要があります。
MB_COMM_LOAD は、いずれかの通信パラメータを変更する必要がある場合にのみ再度呼び出す必要があります。 MB_COMM_LOAD を起動 OB から呼び出して一度実行するか、最初のスキャンシステムフラグを使用し呼び出しを開始して一度実行します。

MB_COMM_LOAD の 1 つのインスタンスを使用して、Modbus 通信に使用する各通信 モジュールの各ポートを設定する必要があります。 使用する各ポートについて一意の MB_COMM_LOAD インスタンスデータブロックを割り付ける必要があります。 S7-1200 CPU は 3 つの通信モジュールに制限されます。

MB_MASTER または MB_SLAVE 命令を書くと、1 つのインスタンスデータブロックが 割り付けられます。 このインスタンスデータブロックは、MB_COMM_LOAD 命令で MB_DB パラメータを指定すると参照されます。

STATUS 値 (W#16#)	説明
0000	エラーは発生していません。
8180	ポートID値が無効です
8181	ボーレート値が無効です
8182	パリティ値が無効です
8183	フロー制御値が無効です
8184	応答タイムアウト値が無効です
8185	MB_MASTER または MB_SLAVE に対するインスタンス DB への正しくない MB_DB ポインタです

6.3.2.2 MB_MASTER



MB_MASTER 命令により、ユーザープログラムは、ポイントツーポイント (PtP) CM 1241 RS485 または CM 1241 RS232 モジュールのポートを使用して、Modbus マスタとして通信を行うことができます。 1 つまたは複数の Modbus スレーブデバイスのデータへのアクセスが可能です。

ユーザープログラムに MB_MASTER 命令を書くと、1 つのインスタンスデータブロックが割り付けられます。 この MB_MASTER インスタンスデータブロック名は、 MB_COMM_LOAD 命令で MB_DB パラメータを指定すると使用されます。

パラメータ	パラメータ タイプ	データタイ プ	説明
REQ	IN	Bool	要求入力: • 0 - 要求なし • 1 - Modbus スレーブへのデータ送信要求
MB_ADR	IN	USInt	Modbus RTU のステーションアドレス: 有効なアドレス範囲: 0~247 値 0 は、すべての Modbus スレーブにメッセージをブロード キャストするために予約されています。 Modbus ファンクションコード 05、06、15、16 のみが、ブロードキャストをサポートしています。
MODE	IN	USInt	モード選択:要求のタイプを指定します。 読み取り、書き込み、または診断 詳細については、以下の Modbus ファンクションの表を参照してください。
DATA_ADDR	IN	UDInt	スレーブの開始アドレス。 Modbus スレーブでアクセスされるデータの開始アドレスを指定します。 有効なアドレスについては、以下の Modbus ファンクションの表を参照してください。

プログラミング手順 6.3 グローバルライブラリ命令

パラメータ	パラメータ タイプ	データタイ プ	説明
DATA_LEN	IN	UInt	データ長: この要求でアクセスされるビット数またはワード数を指定します。 有効な長さについては、以下の Modbusファンクションの表を参照してください。
DATA_PTR	IN	Variant	データポインタ:書き込みまたは読み取り中のデータの CPU DB アドレスをポイントします。 DB は、"シンボリックアクセスのみでない" DB タイプを使用します。 以下の DATA_PTR の注を参照してください。
NDR	OUT	Bool	新規データ有効: • 0 - トランザクションは完了していません • 1 - MB_MASTER 命令が、Modbus スレーブとの要求されたトランザクションを完了しました
BUSY	OUT	Bool	 ビジー: 0 - 進行中の MB_MASTER トランザクションはありません 1 - MB_MASTER トランザクションが進行中です
ERROR	OUT	Bool	 エラー: 0-エラーは検出されませんでした 1-エラーが検出され、パラメータ STATUS のエラーコードは有効です
STATUS	OUT	Word	実行条件コード

Modbus マスタ通信ルール

- MB_MASTER 命令がポートと通信できるようにするには、MB_COMM_LOAD を実行してそのポートを設定する必要があります。
- ポートを使用して Modbus マスタ要求を開始する場合、そのポートを MB_SLAVE で使用することはできません。 そのポートでは、MB_MASTER 実行の 1 つまたは 複数のインスタンスを使用できます。
- Modbus 命令は、通信プロセスの制御に通信割り込みイベントを使用しません。 ユーザープログラムは、完全な条件を送受信するための MB_MASTER 命令をポーリングする必要があります。
- プログラムが Modbus マスタを操作し MB_MASTER を使用して要求をスレーブに 送信する場合、そのスレーブからの要求が返るまでポーリングを続行する (MB_MASTER を実行する)必要があります。
- 同じ OB (または OB 優先度レベル)から、任意のポートに対するすべての MB_MASTER 実行を呼び出します。

REQパラメータ

REQ 値 FALSE = 要求なし

REQ 値 TRUE = Modbus スレーブへのデータ送信要求

この入力は、MB_MASTER 実行の最初の呼び出しで立ち上がりエッジをトリガとした接点を通して供給する必要があります。 立ち上がりエッジによりトリガされたパルスは、送信要求を一度起動します。 すべての入力は、1 つの要求およびこの入力によってトリガされた応答のためにキャプチャされ、変更なしで保持されます。

MB_MASTER は内部的にステートマシンを起動して、この要求が完了するまで、他の MB MASTER 命令を許可しないようにします。

さらに、MB_MASTER FB に対して、要求の完了前に、同じインスタンスの呼び出しが REQ 入力 TRUE で再び実行されると、以降の送信は行われません。 ただし、 MB_MASTER が REQ 入力を True にセットして実行された場合、要求が完了すると、 ただちに新しい要求が発行されます。

DATA_ADDR パラメータと MODE パラメータの Modbus ファンクションタイプの選択

DATA_ADDR (スレーブの開始 Modbus アドレス): Modbus スレーブでアクセスされる データの開始アドレスを指定します。

MB_MASTER は、ファンクションコード入力でなく MODE 入力を使用します。
MODE と Modbus アドレス範囲の組み合わせで、実際の Modbus メッセージで使用されるファンクションコードが決まります。 以下の表は、MBUS_MASTER パラメータ
MODE、Modbus ファンクションコード、Modbus アドレス範囲間の通信を示しています。

MB_MASTER Modbus ファンクション							
	Modbus アドレスパラメータ DATA_ADDR	アドレスタイプ	Modbus データ長パ ラメータ DATA_LEN	Modbus ファンク ション			
モード 0							
読み取り	00001~09999	出力ビット	1~2000	01H			
	10001 – 19999	入力ビット	1~2000	02H			
	30001 - 39999	入力レジスタ	1~125	04H			
	40001~49999	保持レジスタ	1~125	03H			
	400001~465536 (拡張済み)						
モード1							
書き込み	00001~09999	出力ビット	1(シングルビット)	05H			
	40001~49999	保持レジスタ	1(シングルワード)	06H			
	400001~465536 (拡張済み)						
	00001~09999	出力ビット	2~1968	15H			
	40001~49999	保持レジスタ	2~123	16H			
	400001~465536 (拡張済み)						

一部の Modbus スレーブでは、Modbus ファンクション 05H と 06H でシングルビットまたはワードの書き込みをサポートしていません。 この場合、モード 2 が使用され、Modbus ファンクション 15H と 16H を使用してシングルビットおよびワードの書き込みを強制します。

書き込み	00001~09999	出力ビット	1~1968	15H
	40001~49999	保持レジスタ	1~123	16H
	400001~465536 (拡張済み)			

MB_MASTER Modbus ファンクション

モード11

- MB_ADDR への入力として参照される Modbus スレーブからのイベントカウンタワードを読み取ります。
- シーメンス S7-1200 Modbus スレーブ上で、このカウンタは、スレーブが、Modbus マスタからの有効な読み取りまたは書き込み(ブロードキャスト以外)要求を受信するたびに増分します。
- 返される値は、DATA_PTRへの入力として指定されるワード位置に保存されます。
- このモードには有効な DATA_LEN は不要です。

モード80

- MB_ADDR への入力として参照される Modbus スレーブの通信ステータスをチェックします。
- MB_MASTER 命令上の NDR 出力ビットの設定は、アドレス指定された Modbus スレーブが、適切な応答データで応答したことを示します。
- プログラムにデータは返されません。
- このモードには有効な DATA LEN は不要です。

モード81

- MB_ADDR への入力として参照される Modbus スレーブ上のイベントカウンタ(モード 11 により返された)をリセットします。
- MB_MASTER 命令上の NDR 出力ビットの設定は、アドレス指定された Modbus スレーブが、適切な応答データで応答したことを示します。
- プログラムにデータは返されません。
- このモードには有効な DATA LEN は不要です。

DATA_PTR パラメータ

DATA_PTR パラメータは、それぞれ書き込まれる、または読み取られるデータのローカルソースまたは宛先アドレス(S7-1200 CPU 内のアドレス)をポイントします。

MB_MASTER 命令を使用して Modbus マスタを作成する場合、Modbus スレーブとの 読み書き用のデータストレージを提供するグローバルデータブロックを作成する必要が あります。

注記

DATA_PTR パラメータは、[Symbolic Access Only]属性ボックスのチェックを外した状態で作成されたグローバルデータブロックを参照する必要があります。

新規データブロックを追加して、従来のグローバル DB タイプを作成する場合は、 [Symbolic address only]ボックスのチェックを外す必要があります。

DATA_PTR パラメータのデータブロック構造体

- これらのデータタイプは、Modbus アドレス 30001~39999、40001~49999、400001~465536のワード読み取りおよび Modbus アドレス 40001~49999 と400001~465536 へのワード書き込みについて有効です。
 - WORD、UINT、または INT データタイプの標準配列を以下に示します。
 - 名前付き WORD、UINT、または INT 構造体。各要素には一意の名前と 16 ビットデータタイプがあります。
 - 名前付き複合構造体。各要素には一意の名前と **16** または **32** ビットデータタイプ があります。
- ビットの読み書きのための Modbus アドレス 00001~09999 と 10001~19999。
 - ブールデータタイプの標準配列。
 - 一意に命名されたブール変数の名前付きブール構造。
- 必須ではありませんが、各 MB_MASTER 命令にグローバルデータブロックで専用の個別領域を用意することをお勧めします。この推奨事項は、複数のMB_MASTER 命令がグローバルデータブロックの同じ領域に読み書きを行うと、データが破損する可能性が高くなるためです。
- DATA_PTR データ領域が同じグローバルデータブロックに存在する必要はありません。 1 つのデータブロックを Modbus 読み取りの複数領域を使用して作成すること、1 つのデータブロックを Modbus 書き込み用に作成すること、または 1 つのデータブロックを各スレーブステーション用に作成することができます。
- 以下の例のすべての配列は、ベース 1 配列[1 … ##]として作成されます。 配列はベース 0 配列[0 … ###]またはベース 0 とベース 1 の混合として作成することもできます。

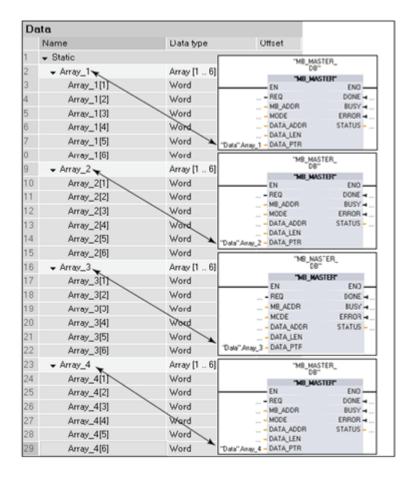
DATA_PTR グローバルデータブロックにアクセスする MB_MASTER 命令の例

以下に示すグローバルデータブロックの例では、Modbus要求データストレージ用に4つの一意の名前と6ワードの配列を使用します。この例のデータ配列のサイズは同じですが、配列は任意のサイズにすることができ、わかりやすくするためにこの例では同じサイズになっています。各配列は、より内容を説明するタグ名や混合データタイプを含むデータ構造体に置き換えることもできます。他のデータ構造体の例は、

「MB_SLAVE命令」 (ページ 281)のHR_DBパラメータの説明に記載されています。

以下の MB_MASTER 命令の例では、DATA_PTR パラメータだけを示していて、他の必 須パラメータは表示していません。 この例の目的は、MB_MASTER 命令がどのように DATA PTR データブロックを使用するかを説明することです。

矢印は、各配列が異なる MB_MASTER 命令に関連付けられている様子を示しています。



どの配列または構造体でも、最初の要素は常に、任意の Modbus 読み取りまたは書き込み動作の最初のソースまたは宛先です。 以下のシナリオは、すべて上の図に基づいています。

シナリオ 1: 最初の MB_MASTER 命令が、任意の有効な Modbus スレーブで Modbus アドレス 40001 から 3 ワードのデータを読み取ると、以下のことが起こります。

アドレス 40001 からのワードが"Data".Array_1[1]に格納されます。

アドレス 40002 からのワードが"Data".Array_1[2]に格納されます。

アドレス 40003 からのワードが"Data".Array_1[3]に格納されます。

シナリオ 2: 最初の MB_MASTER 命令が、任意の有効な Modbus スレーブで Modbus アドレス 40015 から 4 ワードのデータを読み取ると、以下のことが起こります。

アドレス 40015 からのワードが"Data".Array_1[1]に格納されます。

アドレス 40016 からのワードが"Data".Array_1[2]に格納されます。

アドレス 40017 からのワードが"Data".Array_1[3]に格納されます。

アドレス 40018 からのワードが"Data".Array_1[4]に格納されます。

シナリオ 3:2 番目の MB_MASTER 命令が、任意の有効な Modbus スレーブで Modbus アドレス 30033 から 2 ワードのデータを読み取ると、以下のことが起こります。

アドレス 30033 からのワードが"Data".Array_2[1]に格納されます。

アドレス 30034 からのワードが"Data".Array_2[2]に格納されます。

シナリオ 4: 3 番目の MB_MASTER 命令が、任意の有効な Modbus スレーブで Modbus アドレス 40050 から 4 ワードのデータを読み取ると、以下のことが起こります。

"Data".Array_3[1]からのワードが Modbus アドレス 40050 に格納されます。

"Data".Array_3[2]からのワードが Modbus アドレス 40051 に格納されます。

"Data".Array_3[3]からのワードが Modbus アドレス 40052 に格納されます。

"Data".Array_3[4]からのワードが Modbus アドレス 40053 に格納されます。

シナリオ 5: 3 番目の MB_MASTER 命令が、任意の有効な Modbus スレーブで Modbus アドレス 40001 から 3 ワードのデータを読み取ると、以下のことが起こります。

"Data".Array_3[1]からのワードが Modbus アドレス 40001 に格納されます。

"Data".Array_3[2]からのワードが Modbus アドレス 40002 に格納されます。

"Data".Array_3[3]からのワードが Modbus アドレス 40003 に格納されます。

シナリオ 6:4番目の MB_MASTER 命令が、任意の有効な Modbus スレーブから Mode 11 (有効なメッセージカウントを取得)を使用すると、以下のことが起こります。

カウントワードが"Data".Array_4[1]に格納されます。

DATA_PTR 入力としてワード位置を使用したビット読み取りと書き込みの例

表 6-1 シナリオ 7: Modbus アドレス 00001 から開始して 4 つの出力ビットを読み取ります

MB_MASTER 入力值			スレーブ Modbus 値		
MB_ADDR	27 (スレーブ例)		00001	ON	
MODE	0 (読み取り)		00002	ON	
DATA_ADDR	00001 (出力)		00003	OFF	
DATA_LEN	4		00004	ON	
DATA_PTR	"Data".Array_4		00005	ON	
			00006	OFF	
			00007	ON	
			80000	OFF	

Modbus 要求後の"Data".Array_4[1]値					
MS (最上位)バイト	LS (最下位)バイト				
xxxx-1011	xxxx-xxxx				
x はデータが変更されていないことを示します					

表 6-2 シナリオ 8: Modbus アドレス 00003 から開始して 12 の出力ビットを読み取ります

MB_MASTER 入力值		スレーブ	`Modbus 値		
MB_ADDR	27 (スレーブ例)	00001	ON	00010	ON
MODE	0 (読み取り)	00002	ON	00011	OFF
DATA_ADDR	00003 (出力)	00003	OFF	00012	OFF
DATA_LEN	12	00004	ON	00013	ON
DATA_PTR	"Data".Array_4	00005	ON	00014	OFF
		00006	OFF	00015	ON
		00007	ON	00016	ON
		80000	ON	00017	OFF
		00009	OFF	00018	ON

Modbus 要求後の"Data".Array_4[1]値				
MSバイト	LSバイト			
1011-0110	xxxx-0100-			
xはデータが変更されていないことを示します				

表 6-3 シナリオ 9: Modbus アドレス 00001 から開始して 5 の出力ビットを書き込み ます

MB_MASTER 入力值		前のスレーブ出力		後のスレーブ出力
MB_ADDR	27 (スレーブ例)	00001	ON	OFF
MODE	1 (書き込み)	00002	ON	ON
DATA_ADDR	00001 (出力)	00003	OFF	ON
DATA_LEN	5	00004	ON	OFF
DATA_PTR	"Data".Array_4	00005	ON	ON
		00006	OFF	未変更
		00007	ON	未変更
		80000	ON	未変更
		00009	OFF	未変更

Modbus 書き込み要求の"Data".Array_4	4[1]値
MSバイト	LSバイト
xxx1-0110	xxxxx-xxxx
x は Modbus 要求でデータが使用され	ないことを示します

シナリオ 10: Modbus アドレス 00003 から開始して 22 の出力ビットを読み取 表 6-4 ります

MB_MASTER	入力値	スレーフ	ブ Modbus 値			
MB_ADDR	27 (スレーブ例)	00001	ON	00	014	ON
MODE	0(読み取り)	00002	ON	oc	015	OFF
DATA_ADDR	00003 (出力)	00003	OFF	oc	016	ON
DATA_LEN	22	00004	ON	oc	017	ON
DATA_PTR	"Data".Array_4	00005	ON	oc	018	OFF
		00006	OFF	oc	019	ON
		00007	ON	oc	020	ON
		00008	ON	oc	021	OFF
		00009	ON	oc	022	ON
		00010	OFF	oc	023	ON
		00011	OFF	oc	024	OFF
		00012	ON	00	025	OFF
		00013	OFF	00	026	ON

Modbus 要求後の"Data".Array_4[1]値	
MSバイト	LSバイト
0111-0110	0110-1010

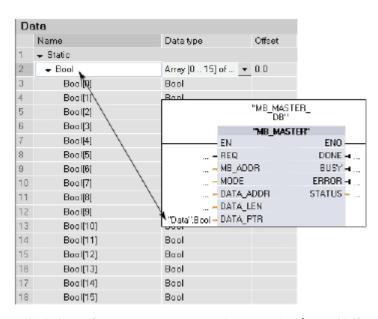
Modbus 要求後の"Data".Array_4[2]値	
MSバイト	LSバイト
xx01-1011	xxxx-xxxx
xはデータが変更されていないことをデ	

DATA_PTR 入力として BOOL 位置を使用したビット読み取りと書き込みの例

ビットアドレス位置への Modbus 読み取りと書き込みはワード位置を使用して処理できますが、DATA_PTR 領域をブールデータタイプ、構造体、または配列として設定して、MB_MASTER 命令を使用して読み取りまたは書き込みされた最初のビットについて、直接的な一対一の相互関係を作ることができます。

ブール領域は、ブール値の配列として、またはブール変数の構造体として作成することができます。 いずれの方法も同じように動作し、相違点はプログラムでの作成および アクセス方法だけです。

以下のグローバルデータブロックエディタの表示は、ベース 0 で作成された 16 ブール値の 1 つの配列を示しています。この配列は、ベース 1 配列としても作成できます。 矢印は、配列が MB MASTER 命令に関連付けられる様子を示しています。



シナリオ 11 と 12 は、Modbus アドレスからブール配列アドレスへの通信を示しています。

表 6-5 シナリオ 11: Modbus アドレス 00001 から開始して 5 の出力ビットを書き込みます

MB_MASTER	R 入力値	前のスレ	ーブ出力	DATA_PTR データ	後のスレーブ出力
MB_ADDR	27 (スレーブ例)	00001	ON	"Data".Bool[0]=FALSE	OFF
MODE	1 (書き込み)	00002	ON	"Data".Bool[1]=TRUE	ON
DATA_ADD	00001 (出力)	00003	OFF	"Data".Bool[2]=TRUE	ON
R					
DATA_LEN	5	00004	ON	"Data".Bool[3]-FALSE	OFF
DATA_PTR	"Data".Bool	00005	ON	"Data".Bool[4]=FALSE	OFF
		00006	OFF		未変更
		00007	ON		未変更
		80000	OFF		未変更

表 6-6 シナリオ 12: Modbus アドレス 00004 から開始して 15 の出力ビットを読み取ります

MB_MASTER 入		スレーブ M	odbus 値	後の DATA_PTR データ
MB_ADDR	27 (スレーブ例)	00001	ON	
MODE	0 (読み取り)	00002	ON	
DATA_ADDR	00003 (出力)	00003	OFF	"Data".Bool[0]=FALSE
DATA_LEN	15	00004	ON	"Data".Bool[1]=TRUE
DATA_PTR	"Data".Bool	00005	ON	"Data".Bool[2]=TRUE
		00006	OFF	"Data".Bool[3]-FALSE
		00007	ON	Data".Bool[4]=TRUE
		80000	ON	Data".Bool[5]=TRUE
		00009	ON	Data".Bool[6]=TRUE
		00010	OFF	Data".Bool[7]=FALSE
		00011	OFF	Data".Bool[8]=FALSE
		00012	ON	Data".Bool[9]=TRUE
		00013	OFF	Data".Bool[10]=FALSE
		00014	ON	Data".Bool[11]=TRUE
		00015	OFF	Data".Bool[12]=FALSE
		00016	ON	Data".Bool[13]=TRUE
		00017	ON	Data".Bool[14]=TRUE
		00018	OFF	
		00019	ON	

条件コード

STATUS 値	説明
(W#16#)	
0000	エラーは発生していません。
80C8	指定された応答タイムアウト(RCVTIME または MSGTIME を参照)が 0 です。
80D1	受信側が、有効な送信を保留するためにフロー制御要求を発行し、指 定された待機時間中に送信を再び有効にしませんでした。
	このエラーは、ハードウェアフロー制御中に、受信側が指定された待機時間内に CTS をアサートしなかった場合にも生成されます
80D2	送信要求は、DCE から DSR 信号が受信されなかったため中止されました。
80E0	受信バッファが不足しているためメッセージが終了しました。
80E1	パリティエラーのためメッセージが終了しました。
80E2	フレーミングエラーのためメッセージが終了しました。
80E3	オーバーランエラーのためメッセージが終了しました。
80E4	指定された長さがバッファサイズ合計を超過しているため、メッセー ジが終了しました。
8180	ポート ID 値が無効です
8186	Modbus ステーションアドレスが無効です
8188	Mode 値または読み取り専用アドレス領域への書き込みモードが無効です
8189	データアドレス値が無効です
818A	データ長値が無効です
818B	• ローカルデータソース/宛先への無効なポインタです: サイズが不正 です
818C	タイプ安全 DB タイプ DATA_PTR へのポインタ(従来の DB タイプであること)
8200	ポートは送信要求の処理中でビジーです

6.3.2.3 MB_SLAVE

MB_SLAVE 命令によって、ユーザープログラムは、ポイントツーポイント(PtP) CM 1241 RS485 または CM 1241 RS232 モジュールのポートを使用して、Modbus スレーブとして通信を行うことができます。 Modbus RTU マスタは要求を発行し、ユーザープログラムは MB_SLAVE の実行により応答します。

ユーザープログラムに MB_SLAVE 命令を書く場合には、1 つの一意のインスタンスデータブロックを割り付ける必要があります。 この MB_SLAVE インスタンスデータブロック名は、MB_COMM_LOAD 命令で MB_DB パラメータを指定すると使用されます。

Modbus 通信ファンクションコード(1、2、4、5、15)は、ビットとワードを PLC 入力 プロセスイメージと出力プロセスイメージで直接読み書きすることができます。 以下 の表は、Modbus アドレスから CPU のプロセスイメージへのマッピングを示しています。

	MB_SLAV	E Modbus >	ファンクシ	ョン		S7-12	00
コード	ファンクション	データエ リア	アドレス	範囲		データエリア	CPUアドレス
01	ビット読 み取り	出力	1	~	8192	出力プロセスイメージ	Q0.0~Q1023.7
02	ビット読 み取り	入力	10001	~	18192	入力プロセスイメージ	I0.0∼I1023.7
04	ワード読 み取り	入力	30001	~	30512	入力プロセスイメージ	IW0∼IW1022
05	ビット書 き込み	出力	1	~	8192	出力プロセスイメージ	Q0.0~Q1023.7
15	ビット書 き込み	出力	1	~	8192	出力プロセスイメージ	Q0.0~Q1023.7

Modbus 通信ファンクションコード(3、6、16)は、個別で一意の Modbus 保持レジスタデータブロックを使用します。 このデータブロックは、MB_SLAVE 命令で MB_HOLD_REG パラメータを指定する前に作成しなければなりません。 以下の表は、 Modbus 保持レジスタから PLC の MB_HOLD_REG DB アドレスへのマッピングを示しています。

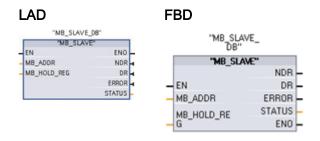
	MB_SLAVE	MB_SLAVE Modbus ファンクション S7-1200			200
コード	ファンクシ	データエ	アドレス範囲	CPU DB データ領域	CPU DB アドレス
	ョン	リア			
03	ワード読み	保持レジ	40001~49999	MB_HOLD_REG	ワード 1~9999
	取り	スタ	400001~465535		ワード 1~65535
06	書き込みワ	保持レジ	40001~49999	MB_HOLD_REG	ワード 1~9999
	ード	スタ	400001~465535		ワード 1~65535
16	ワード書き	保持レジ	40001~49999	MB_HOLD_REG	ワード 1~9999
	込み	スタ	400001~465535		ワード 1~65535

以下の表は、サポートされる Modbus 診断ファンクションを示しています。

S7-1200	MB_SLAVE M	Modbus 診断ファンクション
コード	サブファン	説明
	クション	
08	0000Н	クエリーデータエコーテストを返します。 MB_SLAVE は、受信されたデータの
		ワードを Modbus マスタにエコーバックします。
08	000AH	通信イベントカウンタを消去します。 MB_SLAVE は、Modbus ファンクション
		11 に使用される通信イベントカウンタを消去します。
11		通信イベントカウンタを取得します。 MB_SLAVE は、Modbus スレーブに送信され、成功した読み取りおよび書き込み要求の数を記録する内部通信イベントカウンタを使用します。 カウンタは、ファンクション 8、ファンクション 11、またはブロードキャスト要求では増分しません。 また、通信エラーになった要求
		でも増分されません(パリティエラーや CRC エラーなど)。

MB_SLAVE は、要求が有効な場所へのアクセスである限り、任意の Modbus マスタからのブロードキャスト書き込み要求をサポートします。

要求が有効かどうかに関わらず、MB_SLAVE は、ブロードキャスト要求の結果を Modbus マスタに応答しません。



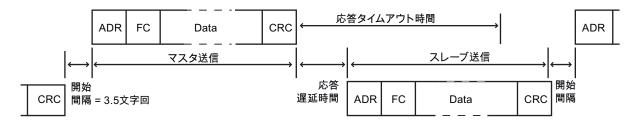
パラメータ	パラメー タタイプ		説明
MB_ADDR	IN	USINT	Modbus RTU アドレス(1~247):
			Modbus スレーブのステーションアドレスで
			す。
MB_HOLD_RE	IN	VARIANT	Modbus 保持レジスタ DB へのポインタ。 保
G			持レジスタ DB は、従来型のグローバル DB
			でなければなりません。以下の
			MB_HOLD_REG の注記を参照してくださ
			٧١°
NDR	OUT	BOOL	新規データ有効:
			0-新規データなし
			• 1 - 新規データが Modbus マスタによって
			書き込まれたことを示します
DR	OUT	BOOL	データ読み取り:
			0-読み取りデータなし
			• 1 - データが Modbus マスタによって読み
			取られたことを示します
ERROR	OUT	BOOL	エラー:
			• 0-エラーは検出されませんでした
			• 1-エラーが検出され、パラメータ
			STATUS のエラーコードは有効です
STATUS	OUT	Word	エラーコード:

Modbus スレーブ通信ルール

- MB_SLAVE 命令がポートと通信できるようにするには、MB_COMM_LOAD を実行してそのポートを設定する必要があります。
- ポートがスレーブとして Modbus マスタに応答する場合、そのポートを MB_MASTER で使用することはできません。 MB_SLAVE 実行の 1 つのインスタン スだけが、任意のポートで使用できます。
- Modbus 命令は、通信プロセスの制御に通信割り込みイベントを使用しません。 ユーザープログラムは、完全な条件を送受信するため MB_SLAVE 命令をポーリングして通信プロセスを制御する必要があります。
- MB_SLAVE は、Modbus マスタからの受信要求に適時に応答できるように、十分な 速度でサイクリック実行する必要があります。
- プログラムサイクル OB からのスキャンごとに MB_SLAVE を呼び出す必要があります。

操作

MB_SLAVE は、Modbus マスタからの各要求を受信し、必要に応じて応答するため、サイクリック実行する必要があります。 MB_SLAVE の実行頻度は、Modbus マスタの 応答タイムアウト時間に依存します。 下図を参照してください。



応答タイムアウト時間は、Modbus マスタが Modbus スレーブの応答開始の待機時間です。 この時間は、Modbus プロトコルによって定義されるのではなく、各 Modbus マスタのパラメータによって定義されます。 MB_SLAVE の実行頻度(ある実行と次の実行との間の時間)は、Modbus マスタの特定のパラメータに基づく必要があります。 最低でも、Modbus マスタの応答タイムアウト時間内で MB_SLAVE を 2 回実行してください。

MB_HOLD_REG パラメータの例

MB_HOLD_REG は、Modbus 保持レジスタデータブロックへのポインタです。 この DB は、Modbus マスタが(読み取りまたは書き込み)アクセスを許可されているデータ値を保持するために使用されます。 このデータブロックを作成し、MB_SLAVE 命令で使用するために、読み取りおよび書き込みが行われるデータタイプの構造体を割り付ける必要があります。

注記

Modbus 保持レジスタデータブロックは、[Symbolic Access Only]属性ボックスのチェックを外した状態で作成されたグローバルデータブロックを参照する必要があります。

新規データブロックを追加して、従来のグローバル DB タイプを作成する場合は、[Symbolic address only]ボックスのチェックを外す必要があります。

保持レジスタは、以下の DB データ構造体を使用できます。

- ワードの標準配列
- 名前付きワード構造体
- 名前付き複合構造体

以下のプログラム例では、MB_HOLD_REG パラメータを使用して DB データ構造体を 処理する方法を説明しています。

例 1 - ワードの標準配列

この例で、保持レジスタはワードの配列です。 データタイプの割付は、他のワードサイズタイプに変更できます(INT および UINT)。

利点:

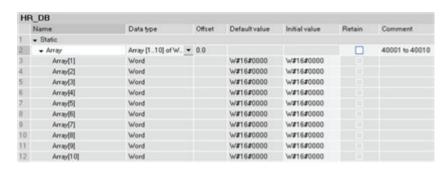
- このタイプの保持レジスタ構造体は、非常に早く簡単に作成できます。
- データ要素にアクセスするプログラムロジックは簡略化されています。

•

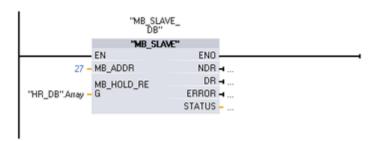
欠点:

- シンボル名により、各配列要素をプログラムによって参照することはできますが("HR_DB"."Array"[1]~"HR_DB"."Array"[10])、 名前によりデータの内部ファンクションはわかりません。
- 配列は 1 つのデータタイプのみでしか構成できません。 タイプ 制御の厳しいユーザープログラムの場合、タイプ変換が必要に なることがあります。

このように、ワード配列構造体がデータブロックエディタに表示されます。



以下の図は、MB_SLAVE 命令の MB_HOLD_REG 入力に配列が割り付けられる様子を示しています。



配列の各要素は、以下のようにシンボル名によってアクセスすることができます。 この例では、新しい値が配列の2番目の要素に移動しています。この要素は、Modbus アドレス40002に対応します。



配列の各ワードは、データブロックで定義されたとおり、MB_SLAVE 命令に Modbus 保持レジスタアドレスを与えます。 この場合、配列には 10 要素しかないため、この MB_SLAVE 命令で使用可能で Modbus マスタがアクセス可能な Modbus 保持レジスタアドレスは、10 しかありません。

配列要素名と Modbus アドレスの相互関係を以下に示します。

"HR_DB".Array[1]	Modbus アドレス 40001
" HR_DB ". ARRAY[2]	Modbus アドレス 40002
" HR_DB ". ARRAY[3]	Modbus アドレス 40003
	•••
" HR_DB ". ARRAY[9]	Modbus アドレス 40009

例2-名前付きワード構造体

この例の保持レジスタは、記述的なシンボル名を持つ一連のワードです。

利点:

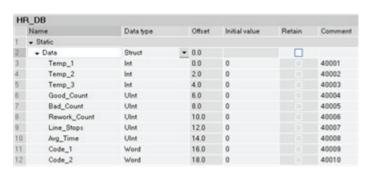
• 各構造体の要素には、固有のデータタイプが割り付けられた記述的な名前があります。

欠点:

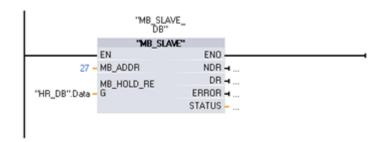
- 標準ワード配列よりも、このタイプの構造体を作成するには長い時間がかかります。
- ユーザープログラムで使用する場合、要素には追加のシンボル 参照が必要です。 単純な配列では最初の要素が

"HR_DB".Array[0]として参照されるのに対し、このタイプの最初の要素は"HR_DB".Data.Temp_1 として参照されます。

こうして、名前付きワード構造体がデータブロックエディタに表示されます。 各要素には一意の名前があり、WORD、UINT、または INT が使用できます。

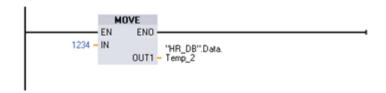


以下の図は、ユーザープログラムで MB_SLAVE 命令の MB_HOLD_REG 入力に上記の データ構造体が割り付けられる様子を示しています。



6.3 グローバルライブラリ命令

配列の各要素は、以下のようにシンボル名によってアクセスすることができます。 この例では、新しい値が配列の 2 番目の要素に移動しています。この要素は、Modbus アドレス 40002 に対応します。



データ要素名と Modbus アドレスの相互関係を以下に示します。

"HR_DB".Data.Temp_1	Modbus アドレス 40001
"HR_DB".Data.Temp_2	Modbus アドレス 40002
"HR_DB".Data.Temp_3	Modbus アドレス 40003
"HR_DB".Data.Good_Count	Modbus アドレス 40004
"HR_DB".Data.Bad_Count	Modbus アドレス 40005
"HR_DB".Data.Rework_Count	Modbus アドレス 40006
"HR_DB".Data.Line_Stops	Modbus アドレス 40007
"HR_DB".Data.Avg_Time	Modbus アドレス 40008
"HR_DB".Data.Code_1	Modbus アドレス 40009
"HR_DB".Data.Code_2	Modbus アドレス 40010

6.3 グローバルライブラリ命令

例3-名前付き複合構造体

この例の保持レジスタは、記述的なシンボル名を持つ一連の混合データタイプです。

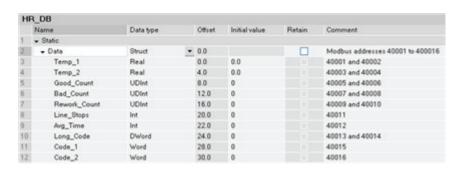
利点:

- 各構造体の要素には、固有のデータタイプが割り付けられた記述的な名前があります。
- これにより、ワードベースではないデータタイプの直接転送が 可能です。

欠点:

- 標準ワード配列よりも、このタイプの構造体を作成するには長い時間がかかります。
- Modbus マスタは、Modbus スレーブから受信するデータを受け付けるように設定する必要があります。 以下の表に示すとおり、Temp_1 は 4 バイトの実数値です。 受信側のマスタは、受信した 2 ワードを予期される実数値に再構築できる必要があります。
- 要素には、ユーザープログラムで追加のシンボル参照が必要になります。 単純な配列では最初の要素が"HR_DB".Array[0]として参照されるのに対し、このタイプの最初の要素は"HR_DB".Data.Temp_1 として参照されます。

こうして、名前付き複合構造体がデータブロックエディタに表示されます。 各要素には、複数のサイズとデータタイプを持つ一意の名前があります。



データ要素名と Modbus アドレスの相互関係を以下に示します。

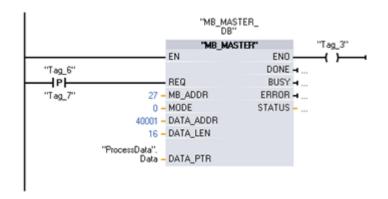
"HR_DB".Data.Temp_1	Modbus アドレス 40001 および 40002
"HR_DB".Data.Temp_2	Modbus アドレス 40003 および 40004
"HR_DB".Data.Good_Count	Modbus アドレス 40005 および 40006
"HR_DB".Data.Bad_Count	Modbus アドレス 40007 および 40008
"HR_DB".Data.Rework_Count	Modbus アドレス 40009 および 40010

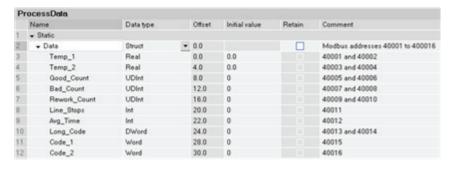
"HR_DB".Data.Line_Stops Modbus アドレス 400011
"HR_DB".Data.Avg_Time Modbus アドレス 400012

"HR_DB".Data.Long_Code Modbus アドレス 40013 および 40014

"HR_DB".Data.Code_1 Modbus アドレス 40015
"HR_DB".Data.Code_2 Modbus アドレス 40016

Modbus マスタとして動作している他の S7-1200 CPU は、MB_Master 命令および同一のデータ構造体を使用して、Modbus スレーブとして動作している S7-1200 CPU からデータブロックを受信することができます。 この Modbus Master 命令は、以下に示すとおり、スレーブの HR_DB データブロックから、マスタの ProcessData データブロックへすべての 16 ワードを直接コピーします。





一連の Modbus マスタ Data_PTR データブロックの場所は、複数の Modbus スレーブ から同じまたは異なる構造体を転送するために使用できます。

6.3 グローバルライブラリ命令

条件コード

STATUS 値 (W#16#)	説明		
80C8	指定された応答タイムアウト(RCVTIME または MSGTIME を参照)が 0 です。		
80D1	受信側が、有効な送信を保留するためにフロー制御要求を発行し、指定された待機時間中に送信を再び有効にしませんでした。 このエラーは、ハードウェアフロー制御中に、受信側が指定された待機時間内に CTS をアサートしなかった場合にも生成されます		
80D2	送信要求は、DCE した	から DSR 信号が受信されなかったため中止されま	
80E0	受信バッファが不	足しているためメッセージが終了しました	
80E1	パリティエラーの	ためメッセージが終了しました	
80E2	フレーミングエラ	ーのためメッセージが終了しました	
80E3	オーバーランエラ	ーのためメッセージが終了しました	
80E4	指定された長さがバッファサイズ合計を超過しているため、メッセー ジが終了しました		
8180	ポート ID 値が無効です		
8186	Modbus ステーションアドレスが無効です		
8187	MB_HOLD_REG DB へのポインタが無効です		
818C	安全 DB タイプ MB_HOLD_REG DB へのポインタ(従来の DB タイプであること)		
	Modbus マスタ に送信された応 答コード (B#16#)		
8380	応答なし CRC エラーです		
8381	01 ファンクションコードがサポートされていません		
8382	応答なし データ長エラーです		
8383	02 データアドレスエラーです		
8384	03 データ値エラーです		
8385	03	データ診断コードの値がサポートされていません(ファンクションコード 08)	

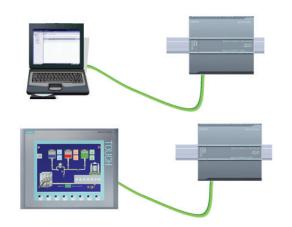
PROFINET

S7-1200 CPU には、イーサネットおよび TCP/IP ベースの通信規格をサポートする統合 された PROFINET ポートがあります。 S7-1200 CPU では以下のアプリケーションプロトコルをサポートしています。

- Transport Control Protocol (TCP)
- ISO on TCP (RFC 1006)

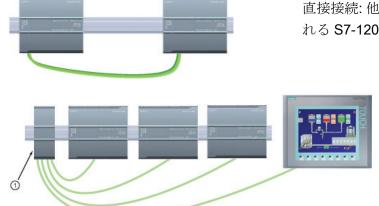
S7-1200 CPU は、他の S7-1200 CPU、STEP 7 Basic プログラミング装置、HMI デバイス、そして標準 TCP 通信プロトコルを使用するシーメンス以外のデバイスと通信を行うことができます。 PROFINET を使用して通信を行う 2 つの方法があります。

- 直接接続: 直接接続は、1 つの CPU に接続されたプログラミング装置、HMI、または他の CPU を使う場合に使用します。
- ネットワーク接続: ネットワーク通信は、3 台以上のデバイスを接続する場合に使用します(CPU、HMI、プログラミング装置、およびシーメンス以外のデバイスなど)。



直接接続: **S7-1200 CPU** に接続される プログラミング装置

直接接続: S7-1200 CPU に接続される HMI



直接接続: 他の S7-1200 CPU に接続される S7-1200 CPU

ネットワーク接続: CSM1277 イーサネットスイッチ①を使用して、3つ以上のデバイスを相互接続します。

イーサネットスイッチは、プログラミング装置または HMI と CPU の直接接続には不要です。 イーサネットスイッチは、3 つ以上の CPU または HMI デバイスのあるネットワークに必要です。 ラックマウントタイプのシーメンス CSM1277 4 ポートイーサネットスイッチを使用して、CPU と HMI デバイスを接続できます。 S7-1200 CPU 上の PROFINET ポートには、イーサネット切り替えデバイスは含まれていません。

PROFINET ポートの最大接続数

CPU の PROFINET ポートは、以下の同時通信接続をサポートします。

- HMI から CPU への通信(3 接続)
- プログラミング装置(PG)から CPU への通信、1接続
- Tブロック命令(TSEND_C、TRCV_C、TCON、TDISCON、TSEN、TRCV)を使用 する S7-1200 プログラム通信、8 接続
- パッシブな S7-1200 CPU とアクティブな S7 CPU の通信、3 接続
 - アクティブ S7 CPU は GET 命令と PUT 命令を使用(S7-300 および S7-400)、または ETHx_XFER 命令(S7-200)を使用します。
 - アクティブ S7-1200 通信接続は、Tブロック命令を使用します。

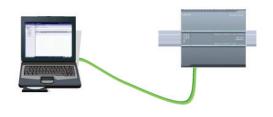
パッシブな ISO および TCP 通信での TSAP の制限

"TCON"命令を使用してパッシブ通信接続を確立すると、以下のポートアドレスが制限され、使用できません。

- ISO TSAP (パッシブ): 01.00, 01.01, 02.00, 02.01, 03.00, 03.01
- TCP ポート(パッシブ): 5001, 102, 123, 20, 21, 25, 34962, 34963, 34964, 80

7.1 プログラミング装置との通信

CPU は、ネットワーク上の **STEP 7** Basic プログラミング装置と通信することができます。



CPU とプログラミング装置間の通信を設定するときは、以下の事項を考慮してください。

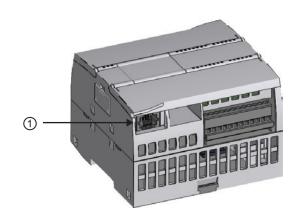
- コンフィグレーション/セットアップ:ハードウェアコンフィグレーションが必要です。
- 一対一の通信でイーサネットスイッチ は不要です。ネットワーク内での3つ 以上のデバイスについてはイーサネットスイッチが必要です。

7.1.1 ハードウェア通信接続の確立

PROFINET インターフェースは、プログラミング装置と CPU 間の物理接続を確立します。 CPU に組み込まれたオートクロスオーバー機能によって、標準またはクロスオーバーイーサネットケーブルを使用できます。 プログラミング装置を CPU に直接接続する場合、イーサネットスイッチは不要です。

以下の手順に従って、プログラミング装置と CPU 間のハードウェア接続を作成します。

- 1. CPUをインストールします (ページ 30)。
- 2. イーサネットケーブルを以下に示す PROFINET ポートに接続します。
- 3. イーサネットケーブルをプログラミング装置に接続します。



① PROFINET ポート

オプションのストレインレリーフを使用して、PROFINET 接続を強化できます。

7.1.2 デバイスのコンフィグレーション

既に CPU とのプロジェクトを作成している場合は、TIA ポータルでプロジェクトを開いてください。

作成していない場合は、プロジェクトを作成し、CPU (ページ 92)をラックに挿入します。 以下のプロジェクトで、CPUはTIAポータルの[Device View]に表示されます。



7.1.3 インターネットプロトコル(IP)アドレスの割り付け

7.1.3.1 IPアドレスのプログラミング装置とネットワークデバイスへの割り付け

プログラミング装置が、プラント LAN に接続されたオンボードアダプタカードを使用している(そして WWW を使用している可能性のある)場合は、CPU とプログラミング装置のオンボードアダプタカードの IP アドレスネットワーク ID およびサブネットマスクは、正確に同じでなければなりません。 ネットワーク ID は、IP アドレスの最初の部分で(最初の3つのオクテット) (たとえば 211.154.184.16)、使用している IP ネットワークを示します。 サブネットマスクは、通常 255.255.255.0 の値ですが、コンピュータがプラント LAN 上にあるため、サブネットマスクの値は、一意のサブネットを設定するため異なる値(たとえば 255.255.254.0)である場合があります。 サブネットマスクをデバイスの IP アドレスに数学の AND 演算で組み合わせると、IP サブネットの境界が定義されます。

注記

プログラミング装置、ネットワークデバイス、IP ルータが世界と通信する WWW では、一意の IP アドレスを割り付けて、他のネットワークユーザーとの衝突を防止します。 IP アドレスの割り付けについては、プラントネットワークに精通した貴社の IT 部門担当者に問い合わせてください。

プログラミング装置が隔離されたネットワークに接続されたイーサネットから USB へのアダプタカードを使用している場合、CPU とプログラミング装置のイーサネットから USB へのアダプタカードの IP アドレスのネットワーク ID およびサブネットマスクは、正確に同じでなければなりません。 ネットワーク ID は、IP アドレスの最初の部分で(最初の3つのオクテット) (たとえば211.154.184.16)、使用している IP ネットワークを示します。 サブネットマスクの値は通常255.255.255.0 です。 サブネットマスクをデバイスの IP アドレスに数学の AND 演算で組み合わせると、IP サブネットの境界が定義されます。

注記

イーサネットから USB アダプタカードの接続は、CPU を企業の LAN に載せたくない場合に便利です。 最初のテストまたは試運転テストで、この準備は特に便利です。

プログラミング装 置のアダプタカー ド		インターネットプロトコル(IP) アドレス	サブネットマスク
オンボードアダプタカード	プラント LAN (および WWW の可能 性のある)接続	CPU とプログラミング装置の オンボードアダプタカードのネットワーク ID は、正確に同じ でなければなりません。 ネットワーク ID は、IP アドレスの最初の部分で(最初の2つのオクテット) (たとえば 211.154.184.16)、使用している IP ネットワークを示します。	CPU とオンボードアダプタカードのサブネットマスクは、全く同じでなければなりません。サブネットマスクは、通常 255.255.255.0 の値ですが、コンピュータがプラント LAN 上にあるため、サブネットマスクの値は、一意のサブネットを設定するため異なる値(たとえば 255.255.254.0)である場合があります。 サブネットマスクをデバイスの IP アドレスに数学の AND 演算で組み合わせると、IP サブネットの境界が定義されます。
イーサネットから USB のアダプタ カード	隔離されたネ ットワークに 接続されてい ます	CPU とプログラミング装置の イーサネットから USB へのア ダプタカードのネットワーク ID は、正確に同じでなければなり ません。 ネットワーク ID は、IP アドレ スの最初の部分で(最初の 2 つ のオクテット) (たとえば 211.154.184.16)、使用している IP ネットワークを示します。	CPU とイーサネットから USB へのアダプタカードのサブネットマスクは、正確に同じでなければなりません。サブネットマスクの値は通常 255.255.255.0 です。サブネットマスクをデバイスの IP アドレスに数学の AND 演算で組み合わせると、IP サブネットの境界が定義されます。

7.1 プログラミング装置との通信

デスクトップの[マイネットワーク]を使用してプログラミング装置の IP アドレスを割り付けまたは確認する

以下のメニュー選択項目によって、プログラミング装置の IP アドレスを割り付けまたは確認できます。

- [(右クリック) [マイネットワーク]
- [プロパティ]
- (右クリック)[ローカルエリア接続]
- [プロパティ]

[ローカルエリアの接続プロパティ]ダイアログの[この接続は次の項目を使用します:] フィールドで、[インターネットプロトコル(TCP/IP)]にスクロールダウンします。 [インターネットプロトコル(TCP/IP)]をクリックし、[プロパティ]ボタンをクリックします。 [IP アドレスを自動取得する(DHCP)]または[次の IP アドレスを使う:](スタティック IP アドレスを入力する)を選択します。

注記

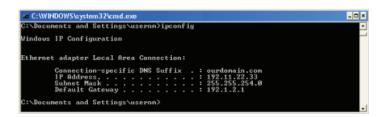
DHCP サーバーからプログラミング装置に電源が入ると、ダイナミックホストコンフィグレーションプロトコル(DHCP)が、自動的に IP アドレスを割り付けます。

プログラミング装置の IP アドレスを確認するには、"ipconfig"コマンドと"ipconfig /all"コマンドを使用します。

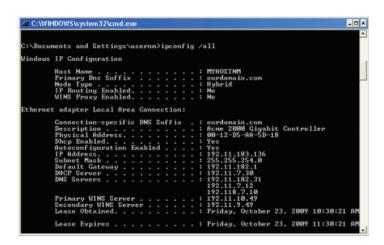
また、プログラミング装置のIPアドレスは、当てはまる場合に次のメニュー選択項目によって、IPルータ(ゲートウェイ)のIPアドレスも確認することができます。

- [スタート]ボタン(デスクトップ上)
- [ファイル名を指定して実行]

[ファイル名を指定して実行]ダイアログの[開く]フィールドに、"cmd"を入力し[OK]ボタンをクリックします。 表示される[C:\WINDOWS\system32\cmd.exe]ダイアログで、コマンド"ipconfig"をクリックします。 結果の例を以下に示します。



詳細情報は、"ipconfig /all"コマンドを使用して表示できます。 プログラミング装置のアダプタカードタイプとイーサネット(MAC)アドレスはここに表示されます。



IP アドレスの CPU への割り付け

以下の2つの方法のいずれかを使って、IP アドレスをCPU に割り付けることができます。

- IP アドレスをオンラインで割り付ける
- IP アドレスをプロジェクトで設定する

7.1.3.2 IPアドレスのオンラインでの割り付け

IP アドレスをネットワークデバイスにオンラインで割り付けることができます。 これは、初期デバイスコンフィグレーションで特に便利です。

以下の手順を使用して、IPアドレスをオンラインで割り付けます。

- 1. [Project tree]で、CPU に IP アドレスが割り付けられていないことを次のメニュー選択項目で確認します。
- · [Online access]
- <デバイスのあるネットワーク用アダプタカード>
- [Update accessible devices]
- 2. [Project tree]で、次のメニュー項目を選択します。
- [Online access]
- <デバイスのあるネットワーク用アダプタカード>
- [Update accessible devices]
- <デバイスアドレス>
- [Online & diagnostics]



Print...
Print preview...

Properties.

Ctrl+P

Alt+Enter

7.1 プログラミング装置との通信

Paddress: 192 . 168 . 0 . , 1 bnet screen mask: 255 . 255 . 255 . 0 re router: 192 . 168 . 0 . . 1

- 3. [Online & diagnostics]ダイア ログで、次のメニュー項目を選 択します。
- [Functions]
- [Assign IP address]



agnostics
General
Diagnostics status
Standard diagnostics
Cycle time

- 4. [IP address]フィールドに、 新しい IP アドレスを入力しま す。
- 5. [Project tree]で、CPU に新し い IP アドレスが割り付けられ ていることを次のメニュー選択 項目で確認します。
- [Online access]
- <デバイスのあるネットワー ク用アダプタ>
- [Update accessible devices]

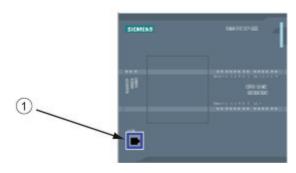


7.1 プログラミング装置との通信

7.1.3.3 IPアドレスをプロジェクトで設定する

PROFINET インターフェースを設定する

CPUをラック上に構成 (ページ 297)したら、PROFINETインターフェース用のパラメータを設定することができます。 CPU上の緑色の[PROFINET]ボックスをクリックし、 [PROFINET port]を選択します。 インスペクタウィンドウの[Properties]タブに、 PROFINETポートが表示されます。



① PROFINET ポート

IP アドレスを設定する

イーサネット(MAC)アドレス: PROFINET ネットワークの各デバイスには、製造メーカーによって識別用のメディアアクセス制御アドレス(MAC アドレス)が割り付けられています。 MAC アドレスは、2 桁の 16 進数を伝送順にハイフン(-)またはコロン(:)で区切った 6 つのグループで構成されています(01-23-45-67-89-AB や 01:23:45:67:89:AB など)。

IP address: また、各デバイスにはインターネットプロトコル(IP)アドレスも必要です。 このアドレスを使用して、デバイスはさらに複雑にルーティングされたネットワーク上 にデータを配信することができます。

IP アドレスは 8 ビットのセグメントに分割され、ドット付き 10 進形式で表されます (211.154.184.16 など)。 IP アドレスの最初の部分はネットワーク ID (ネットワークの 識別)で、2 番目の部分はホスト ID (ネットワーク上のそれぞれのデバイスに固有)です。 IP アドレス 192.168.x.y は、インターネット上でルーティングされていないプライベートネットワークの一部として認識されるアドレスの標準指定です。

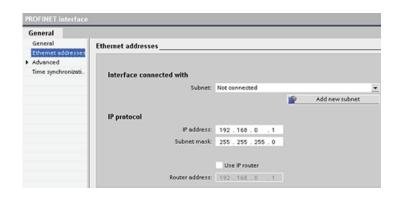
Subnet mask: サブネットは、接続されているネットワークデバイスの論理的な分類です。 サブネット上のノードは、通常は、ローカルエリアネットワーク(LAN)上の相互に物理的に近い場所に配置されます。 マスク(サブネットマスクまたはネットワークマスクとも呼ばれる)は、IP サブネットの境界を定義します。

サブネットマスク 255.255.255.0 は、一般的に小規模ローカルネットワークに適しています。 つまり、このネットワーク上のすべての IP アドレスは最初の 3 オクテットが同じで、このネットワーク上のさまざまなデバイスは最後のオクテット(8 ビットのフィールド)で識別されます。 たとえば、サブネットマスク 255.255.255.0 と IP アドレス 192.168.2.0~192.168.2.255 を小規模ローカルネットワーク上のデバイスに割り付けます。

異なるサブネット間の接続のみがルータを経由します。 サブネットを使用する場合は、IP ルータが必要です。

IP router: ルータは LAN 間を接続します。 LAN 内のコンピュータは、ルータを使用して、他のネットワークにメッセージを送信することができます。そのネットワークの背後に、また他の LAN が存在する可能性もあります。 データの送信先が LAN 内に存在しない場合、ルータは、そのデータを送信先に配信することができる他のネットワークまたはネットワークのグループに転送します。

ルータは IP アドレスを使用して、データパケットの送受信を行います。



IP addresses プロパティ: [Properties]ウィンドウで[Ethernet address]設定エントリを選択します。 TIA ポータルに [Ethernet address configuration]ダイアログが表示されます。このダイアログで、ソフトウェアプロジェクトを受信する CPUの IP アドレスの関連付けを行います。

7.1 プログラミング装置との通信

注記

CPUのIPアドレスは未設定です。IPアドレスを手動でCPUに割り付ける必要があります。CPUがネットワーク上のルータに接続されている場合は、そのルータのIPアドレスも入力する必要があります。プロジェクトをダウンロードすると、すべてのIPアドレスが設定されます。

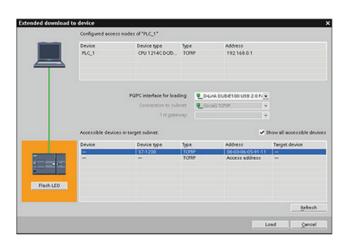
詳細については、「プログラミングデバイスおよびネットワークデバイスに IP アドレスを割り付ける」を参照してください。

IPアドレス用パラメータの定義を下の表に示します。

パラメータ	説明	
Subnet	デバイスの接続先サブネット名。 新しいサブネットを作成するには[Add new subnet]ボタンをクリックする。 [Not connected]がデフォルト。 次の 2 種類の接続タイプが指定できる。 ・ デフォルトの[Not connected]によるローカル接続 ・ ネットワーク上に 2 つ以上のデバイスが存在する場合は、サブネットが必要	
IP protocol	IP address	CPU に割り付ける IP アドレス
	Subnet mask	割り付けるサブネットマスク
	Use IP router	IP ルータを使用するには、このチェックボックスを オンにする
	Router address	ルータを使用する場合にルータに割り付ける IP アドレス

7.1.4 PROFINETネットワークのテスト

コンフィグレーションを完了した後、CPU にプロジェクトをダウンロードします。 プロジェクトをダウンロードすると、すべての IP アドレスが設定されます。



IP アドレスのオンラインでのデバイスへの割り付け

S7-1200 CPU には、事前設定された IP アドレスはありません。 IP アドレスを手動で CPU に割り付ける必要があります。

IP アドレスをオンラインでデバイスに割り付けるには、段階的な手順を「IP アドレスをオンラインで割り付ける」で参照してください。

IPアドレスをプロジェクトに割り付けるには、IPアドレスをデバイスのコンフィグレーションで設定し、そのコンフィグレーションを保存して、PLCにダウンロードする必要があります。 詳細については、「IPアドレスをプロジェクトに設定する」を参照してください。

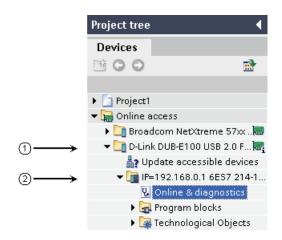
注記

IP アドレスをオンラインで割り付けた場合、オンラインまたはオフラインのハードウェアコンフィグレーション方法を使用して、オンラインで割り付けた IP アドレスを変更できます。

IPアドレスをオフラインで割り付けた場合、オフラインのハードウェアコンフィグレーション方法を使用してのみ、プロジェクトに割り付けた IPアドレスを変更できます。

7.1 プログラミング装置との通信

以下のように[Online access]を使用して、接続された CPU の IP アドレスを表示します。



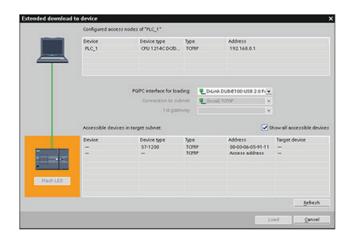
- ① このプログラミング装置に関する2つのイーサネットワークの2番目
- ② このイーサネットワーク上にある唯一の S7-1200 CPU の IP アドレス

注記

プログラミング装置のすべての設定済みネットワークが表示されます。 必要な **S7-1200 CPU** の **IP** アドレスを表示するには、正しいネットワークを選択する必要があります。

[Extended download to device]ダイアログを使用して接続されたネットワークデバイスをテストする

S7-1200 CPU の[Download to device] ファンクションとその[Extended download to device] ダイアログで、すべてのアクセス可能なネットワークデバイスを表示し、すべてのデバイスに一意の IP アドレスが割り付けられているかどうかを確認することができます。 すべてのアクセス可能で使用可能なデバイスをその割り付けられた MAC アドレスと IP アドレスと共に表示するには、[Show all accessible devices] チェックボックスをオンにします。



必要なネットワークデバイスがリストにない場合は、何らかの理由でそのデバイスへの 通信が中断されています。 このデバイスとネットワークに、ハードウェアエラーやコ ンフィグレーションエラーがないかどうかを調査する必要があります。

7.2 HMI から PLC への通信



CPU は、HMI への PROFINET 通信接続をサポートしています。 CPU と HMI 間の通信を設定する場合は、以下の要件を考慮する必要があります。

コンフィグレーション/セットアップ:

- CPUの PROFINET ポートは、HMI と接続するように設定されている必要があります。
- HMI はセットアップおよび設定する必要があります。
- HMI コンフィグレーション情報は CPU プロジェクトの一部であり、プロジェクト でのコンフィグレーションとダウンロードが可能です。
- 一対一の通信でイーサネットスイッチは不要です。ネットワーク内での**3**つ以上の デバイスについてはイーサネットスイッチが必要です。

注記

ラックマウントタイプのシーメンス CSM1277 4 ポートイーサネットスイッチを使用して、CPU と HMI デバイスを接続できます。 CPU 上の PROFINET ポートには、イーサネット切り替えデバイスは含まれていません。

サポートされているファンクション:

- HMI は CPU との間でデータの読み書きができます。
- **CPU** から取得した情報に基づいてメッセージをトリガできます。
- システム診断

注記

WinCC Basic および STEP 7 Basic は TIA ポータルのコンポーネントです。 HMI の コンフィグレーションについての詳細は、WinCC Basic を参照してください。

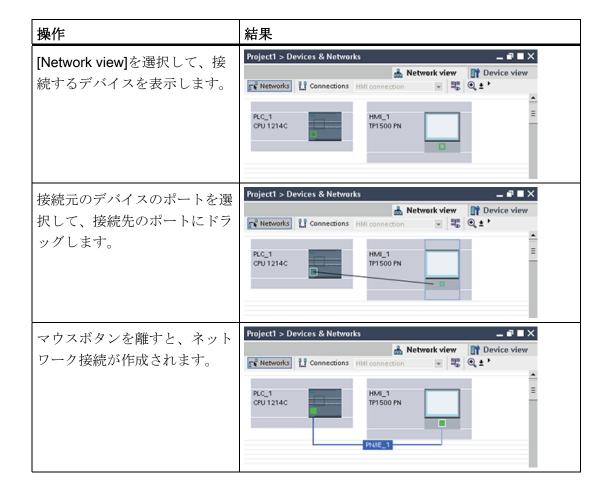
HMI と CPU 間の通信を設定するために必要な手順

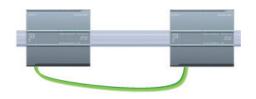
手順	タスク
1	ハードウェア通信接続の確立
	PROFINET インターフェースが、HMI と CPU 間の物理接続を確立します。
	CPU に組み込まれたオートクロスオーバー機能によって、標準またはクロスオ
	ーバーイーサネットケーブルをインターフェースに使用できます。 HMI と
	CPU の接続にイーサネットスイッチは不要です。
	「プログラミング装置との通信」を参照してください。 詳細については、「ハードウェア通信の接続の確立」 (ページ 296) を参照してください。
2	デバイスのコンフィグレーション
	「プログラミング装置との通信」を参照してください。 詳細については、「デバイスのコンフィグレーション」 (ページ 297)を参照してください。
3	HMI と CPU 間の論理ネットワーク接続のコンフィグレーション
	「HMIからPLCへの通信」を参照してください。 詳細については、「HMIと
	CPU間の論理ネットワーク接続のコンフィグレーション」 (ページ 312)を参照してください。
4	IP アドレスをプロジェクトで設定する
•	プー・ブラン・マニテー くいだ
	する必要があります。
	「プログラミング装置との通信」を参照してください。 詳細については、「IP
	アドレスをプロジェクトで設定する」 (ページ 304)を参照してください。
5	PROFINET ネットワークのテスト
	各 CPU にコンフィグレーションをダウンロードする必要があります。
	「プログラミング装置との通信」を参照してください。 詳細については、
	「PROFINETネットワークのテスト」 (ページ 307)を参照してください。

7.2.1 HMIとCPU間の論理ネットワーク接続のコンフィグレーション

CPU とラックを設定したら、ネットワーク接続の設定を開始できます。

[Devices and Networks]ポータルで[Network view]を使用して、プロジェクト内のデバイス間でネットワーク接続を作成します。 イーサネット接続を作成するには、CPU で緑の(イーサネット)ボックスを選択します。 ラインをイーサネットボックスの HMI デバイスにドラッグします。 マウスのボタンを離すと、イーサネット接続が作成されます。





CPU は、**TSEND_C** 命令と **TRCV_C** 命令を使用して、ネットワーク上の別の **CPU** と通信することができます。

2つの CPU 間の通信を設定する場合は、以下の事項を考慮してください。

- コンフィグレーション/セットアップ: ハードウェアコンフィグレーションが必要です。
- サポートされているファンクション: CPU とのデータの読み書き
- 一対一の通信でイーサネットスイッチは不要です。ネットワーク内での3つ以上の デバイスについてはイーサネットスイッチが必要です。

2つの CPU 間での通信のコンフィグレーションに必要な手順

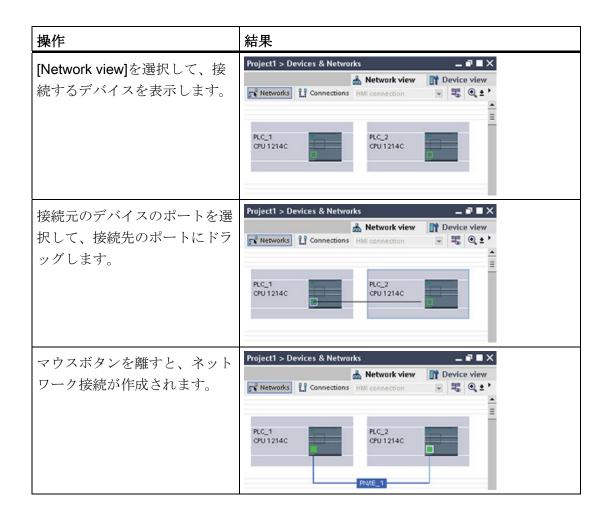
手順	タスク
1	ハードウェア通信接続の確立
	PROFINET インターフェースは、2 つの CPU 間の物理接続を確立します。
	CPU に組み込まれたオートクロスオーバー機能によって、標準またはクロスオ
	ーバーイーサネットケーブルをインターフェースに使用できます。 2 つの CPU
	間の接続にイーサネットスイッチは不要です。
	「プログラミング装置との通信」を参照してください。 詳細については、「ハ
	ードウェア通信の接続の確立」を参照してください。
2	デバイスのコンフィグレーション
	2 つのプロジェクトを設定する必要があります(各プロジェクトに 1 つの
	CPU) _o
	「プログラミング装置との通信」を参照してください。 詳細については、「デ
	バイスのコンフィグレーション」を参照してください。
3	2つの CPU 間の論理ネットワーク接続のコンフィグレーション
	「2つのCPU間の通信のコンフィグレーション」を参照してください。 詳細に
	ついては、「2つのCPU間の論理ネットワーク接続のコンフィグレーション」
	(ページ 315)を参照してください。

手順	タスク
4	IP アドレスをプロジェクトで設定する
	同じコンフィグレーションプロセスを使用しますが、2 つの CPU に IP アドレ
	スを設定する必要があります(PLC_1 および PLC_2 など)。
	「プログラミング装置との通信」を参照してください。 詳細については、「IP
	アドレスをプロジェクトで設定する」を参照してください。
5	送信パラメータと受信パラメータのコンフィグレーション
	2つの CPU 間での通信を可能にするには、両方の CPU で TSEND_C 命令と
	TRCV_C 命令を設定する必要があります。
	「2つのCPU間の通信のコンフィグレーション」を参照してください。 詳細に
	ついては、「送信パラメータと受信パラメータのコンフィグレーション」 (ペ
	ージ 315)を参照してください。
6	PROFINET ネットワークのテスト
	各 CPU にコンフィグレーションをダウンロードする必要があります。
	「プログラミング装置と CPU 間の通信のコンフィグレーション」を参照して
	ください。 詳細については、「PROFINET ネットワークのテスト」を参照し
	てください。

7.3.1 2 つのCPU間の論理ネットワーク接続のコンフィグレーション

CPU とラックを設定したら、ネットワーク接続の設定を開始できます。

[Devices and Networks]ポータルで[Network view]を使用して、プロジェクト内のデバイス間でネットワーク接続を作成します。 PROFINET 接続を作成するには、最初の PLC で緑の(PROFINET)ボックスを選択します。 ラインを PROFINET ボックスの PLC デバイスにドラッグします。 マウスのボタンを離すと、PROFINET 接続が作成されます。



7.3.2 送信パラメータと受信パラメータのコンフィグレーション

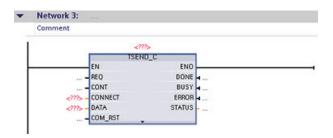
2つの CPU 間の接続を確立するには、送信ブロック(T ブロック)通信を使用します。 CPU が PROFINET 通信を開始できるようにするには、メッセージの送信パラメータと 受信パラメータを設定する必要があります。 これらのパラメータにより、ターゲット 装置とのメッセージの送受信での通信動作が決まります。

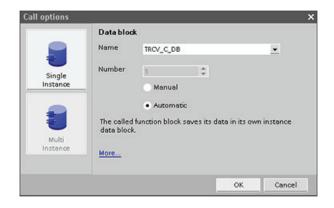
7.3.2.1 TSEND_C命令の送信パラメータのコンフィグレーション

TSEND_C 命令

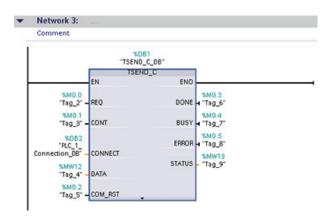
TSEND_C命令 (ページ 212)は、パートナーステーションへの通信接続を作成します。 セットアップおよび確立された接続は、命令によって切断を指示されるまで自動的にモニタされます。 TSEND_C命令は、TCON、TDISCON、TSEND命令のファンクションを組み合わせたものです。

STEP 7 Basic のデバイスコンフィグレーションから、TSEND_C 命令がデータを送信する方法を設定できます。 開始するには、[Extended Instructions]の[Communications] フォルダからプログラムに命令を挿入します。 命令は、[Call]オプションダイアログと共に表示されます。このダイアログで、TSEND_C 命令のパラメータを格納する DB を割り付けます。





以下の図に示すように、タグメモリの入力および出力への位置を割り付けることができます。



一般的なパラメータのコンフィグレーション

TSEND_C 命令の[Properties]設定ダイアログで、通信パラメータを指定します。 この ダイアログは、TSEND_C 命令の任意の部分を選択すると、常にページ下部付近に表示 されます。

接続パラメータのコンフィグレーション

各 CPU には統合された PROFINET ポートがあり、これが標準 PROFINET 通信をサポートします。 サポートされているイーサネットプロトコルは、次の 2 つの接続タイプで、以下に説明します。

プロト コル	プロトコル名	使用方法
RFC 1006	ISO on TCP	メッセージのフラグメント化と再構築
TCP	Transport Control Protocol	フレームの転送

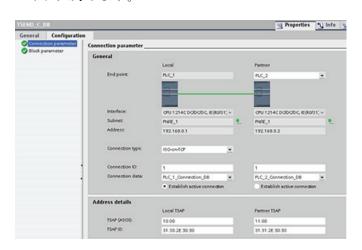
ISO on TCP (RFC 1006)

ISO on TCP は、ISO アプリケーションを TCP/IP ネットワークにポートできるように するメカニズムです。 このプロトコルには、以下の特長があります。

- ハードウェアに緊密に接続された効率的な通信プロトコルです。
- 中規模から大規模のデータ量に適しています(最大 8192 バイト)。
- TCP と対照的に、メッセージはデータの終わりでの識別が特徴で、メッセージ指向です。
- ルーティングが可能で、WANで使用できます。
- ダイナミックデータ長が可能です。
- データ管理には、SEND / RECEIVE プログラミングインターフェースによってプログラミング作業が必要です。

トランスポートサービスアクセスポイント(TSAP)を使用して、TCP プロトコルで複数の接続を 1 つの IP アドレスに関連付けることができます(最大 64K 接続)。 RFC 1006 により、TSAP は 1 つの IP アドレスへの複数のエンドポイント接続を一意に識別します。

[Connection Parameters]ダイアログの[Address Details]セクションで、使用する TSAP を定義することができます。 CPU での接続 TSAP は、[Local TSAP]フィールドに入力します。 パートナーCPU での接続に割り付けられる TSAP は、[Partner TSAP]フィールドに入力します。



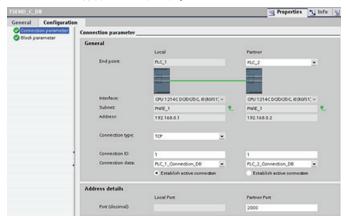
パラメータ	定義
全般	
エンドポイント: パー	パートナー(受信側) CPU に割り付ける名前
トナー	
インターフェース	インターフェースに割り付ける名前
Subnet	サブネットに割り付ける名前
アドレス	割り付けられた IP アドレス
接続タイプ	イーサネットプロトコルのタイプ
接続 ID	ID番号
接続データ	ローカルおよびパートナーの CPU データ保管場所
アクティブな接続の	ローカルまたはパートナーCPU をアクティブな接続として選
セットアップ	択するラジオボタン
アドレスの詳細	
TSAP1 (ASCII)	ローカルおよびパートナーの CPU TSAP (ASCII フォーマット)
TSAP ID	ローカルおよびパートナーの CPU TSAP (16 進フォーマット)

¹ S7-1200 CPU に ISO on TCP の接続を設定する場合は、パッシブ通信パートナーの TSAP 拡張には ASCII 文字だけを使用してください。

Transport Control Protocol (TCP)

TCP は RFC 793: Transmission Control Protocol で記述される標準プロトコルです。 TCP の主な目的は、信頼性のある安全な接続サービスをプロセスペア間で提供することです。 このプロトコルには、以下の特長があります。

- ハードウェアに緊密に接続された効率的な通信プロトコルです。
- 中規模から大規模のデータ量に適しています(最大 8192 バイト)。
- 特に以下のアプリケーションに、さらに多くの機能を提供します。
 - エラー回復
 - フロー制御
 - 信頼性
- コネクション型プロトコル
- TCP だけをサポートするサードパーティシステムと共に非常に高い柔軟性を持って 使用できます。
- ルーティング機能あり
- スタティックデータ長のみ適用可能です。
- メッセージは応答されます。
- アプリケーションはポート番号を使用してアドレス指定されます。
- TELNET、FTP などのほとんどのユーザーアプリケーションプロトコルは TCP を使用します。
- データ管理には、SEND / RECEIVE プログラミングインターフェースによってプログラミング作業が必要です。



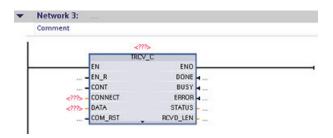
パラメータ	定義
全般	
エンドポイント: パートナー	パートナー(受信側) CPU に割り付ける名前
インターフェース	インターフェースに割り付ける名前
Subnet	サブネットに割り付ける名前
アドレス	割り付けられた IP アドレス
接続タイプ	イーサネットプロトコルのタイプ
接続 ID	ID番号
接続データ	ローカルおよびパートナーの CPU データ保管場所
アクティブな接続の セットアップ	ローカルまたはパートナー CPU をアクティブな接続として選択 するラジオボタン
アドレスの詳細	
Port (10 進)	10 進フォーマットのパートナーCPU ポート

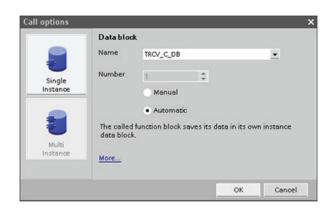
7.3.2.2 TRCV_C命令の受信パラメータのコンフィグレーション

TRCV_C 命令

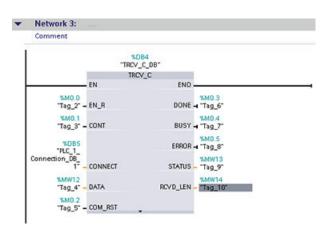
TRCV_C命令 (ページ 212)は、パートナーステーションへの通信接続を作成します。 セットアップおよび確立された接続は、命令によって切断を指示されるまで自動的にモニタされます。 TRCV_C命令は、TCON、TDISCON、TRCV命令のファンクションを組み合わせたものです。

STEP 7 Basic の CPU コンフィグレーションから、TRCV_C 命令がデータを受信する 方法を設定できます。 開始するには、[Extended Instructions]の[Communications]フォルダからプログラムに命令を挿入します。 命令は、[Call]オプションダイアログと共に 表示されます。このダイアログで、TRCV_C 命令のパラメータを格納する DB を割り付けます。





以下の図に示すように、タグメモリの入力および出力への位置を割り付けることができます。



一般的なパラメータのコンフィグレーション

TRCV_C 命令の[Properties]コンフィグレーションダイアログで、通信パラメータを指定します。 このダイアログは、TRCV_C 命令の任意の部分を選択すると、常にページ下部付近に表示されます。

接続パラメータのコンフィグレーション

各 CPU には統合された PROFINET ポートがあり、これが標準 PROFINET 通信をサポートします。 サポートされているイーサネットプロトコルは、次の 2 つの接続タイプで、以下に説明します。

プロトコル	プロトコル名	使用方法
RFC 1006	ISO on TCP	メッセージのフラグメント化と再構築
TCP	Transport Control Protocol	フレームの転送

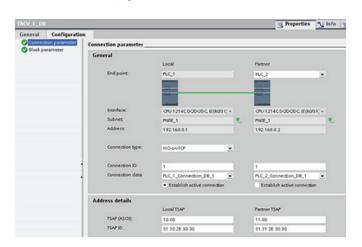
ISO on TCP (RFC 1006)

ISO on TCP は、ISO アプリケーションを TCP/IP ネットワークにポートできるように するメカニズムです。 このプロトコルには、以下の特長があります。

- ハードウェアに緊密に接続された効率的な通信プロトコルです。
- 中規模から大規模のデータ量に適しています(最大 8192 バイト)。
- TCP と対照的に、メッセージはデータの終わりでの識別が特徴で、メッセージ指向です。
- ルーティングが可能で、WANで使用できます。
- ダイナミックデータ長が可能です。
- データ管理には、SEND / RECEIVE プログラミングインターフェースによってプログラミング作業が必要です。

トランスポートサービスアクセスポイント(TSAP)を使用して、TCP プロトコルで複数 の接続を 1 つの IP アドレスに関連付けることができます(最大 64K 接続)。 RFC 1006 により、TSAP は 1 つの IP アドレスへの複数のエンドポイント接続を一意に識別します。

[Connection Parameters]ダイアログの[Address Details]セクションで、使用する TSAP を定義することができます。 CPU での接続 TSAP は、[Local TSAP]フィールドに入力します。 パートナーCPU での接続に割り付けられる TSAP は、[Partner TSAP]フィールドに入力します。



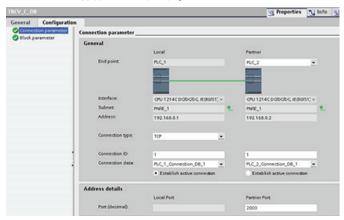
パラメータ	定義
全般	
エンドポイント: パートナー	パートナー(受信側) CPU に割り付ける名前
インターフェース	インターフェースに割り付ける名前
Subnet	サブネットに割り付ける名前
アドレス	割り付けられた IP アドレス
接続タイプ	イーサネットプロトコルのタイプ
接続 ID	ID番号
接続データ	ローカルおよびパートナーの CPU データ保管場所
アクティブな接続の セットアップ	ローカルまたはパートナー CPU をアクティブな接続として選択 するラジオボタン
アドレスの詳細	
TSAP1 (ASCII)	ローカルおよびパートナーの CPU TSAP (ASCII フォーマット)
TSAP ID	ローカルおよびパートナーの CPU TSAP (16 進フォーマット)

¹ S7-1200 CPU に ISO on TCP の接続を設定する場合は、パッシブ通信パートナーの TSAP 拡張には ASCII 文字だけを使用してください。

Transport Control Protocol (TCP)

TCP は RFC 793: Transmission Control Protocol で記述される標準プロトコルです。 TCP の主な目的は、信頼性のある安全な接続サービスをプロセスペア間で提供することです。 このプロトコルには、以下の特長があります。

- ハードウェアに緊密に接続された効率的な通信プロトコルです。
- 中規模から大規模のデータ量に適しています(最大 8192 バイト)。
- 特に以下のアプリケーションに、さらに多くの機能を提供します。
 - エラー回復
 - フロー制御
 - 信頼性
- コネクション型プロトコル
- TCP だけをサポートするサードパーティシステムと共に非常に高い柔軟性を持って 使用できます。
- ルーティング機能あり
- スタティックデータ長のみ適用可能です。
- メッセージは応答されます。
- アプリケーションはポート番号を使用してアドレス指定されます。
- TELNET、FTP などのほとんどのユーザーアプリケーションプロトコルは TCP を使用します。
- データ管理には、SEND / RECEIVE プログラミングインターフェースによってプログラミング作業が必要です。



パラメータ	定義
全般	
エンドポイント: パートナー	パートナー(受信側) CPU に割り付ける名前
インターフェース	インターフェースに割り付ける名前
Subnet	サブネットに割り付ける名前
アドレス	割り付けられた IP アドレス
接続タイプ	イーサネットプロトコルのタイプ
接続 ID	ID番号
接続データ	ローカルおよびパートナーの CPU データ保管場所
アクティブな接続の セットアップ	ローカルまたはパートナー CPU をアクティブな接続として選択 するラジオボタン
アドレスの詳細	
Port (10 進)	10 進フォーマットのローカル CPU ポート

7.4 リファレンス情報

7.4.1 CPU上のイーサネット(MAC)アドレスの検索

PROFINET ネットワーキングで、Media Access Control アドレス(MAC アドレス)は、メーカーがアダプタカードに割り付けた識別子です。 通常 MAC アドレスは、メーカーの登録識別番号をエンコードしています。

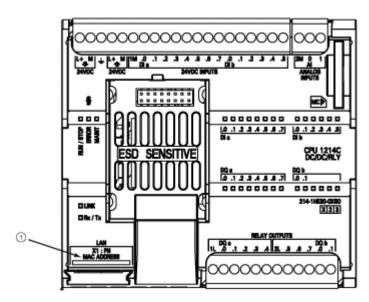
MAC アドレス表示の標準(IEEE 802.3)フォーマットはヒューマンフレンドリな 2 つの 16 進数の 6 つのグループで、ハイフン(-)またはコロン(:)で送信順序によって区切られています(例えば、01-23-45-67-89-ab や 01:23:45:67:89:ab)。

注記

各 CPU には、工場設定の永続的な一意の MAC アドレスがロードされます。 CPU の MAC アドレスを変更することはできません。

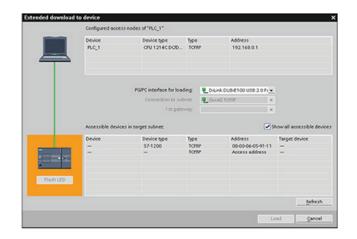
7.4 リファレンス情報

MAC アドレスは、CPU の左下隅前に印刷されています。 MAC アドレス情報を確認するには、下側のフロントカバーを上げる必要があります。



① MAC アドレス

最初、CPU には工場でインストールされた MAC アドレス以外に IP アドレスはありません。 PROFINET 通信では、すべてのデバイスに一意の IP アドレスを割り付ける必要があります。



CPU の[Download to device]フ ァンクションとその[Extended download to device]ダイアログ で、すべてのアクセス可能な ネットワークデバイスを表示 し、すべてのデバイスに一意 の IP アドレスが割り付けられ ているかどうかを確認するこ とができます。このダイアロ グには、すべてのアクセス可 能で利用可能なデバイスが、 割り付けられた MAC アドレス とIPアドレスと共に表示され ます。 MAC アドレスは、必要 な一意の IP アドレスのないデ バイスを識別するため非常に 重要です。

7.4.2 ネットワークタイムプロトコル同期化のコンフィグレーション

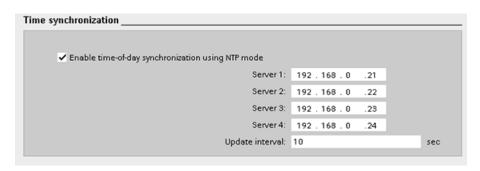
ネットワークタイムプロトコル(NTP)は、コンピュータシステムのクロックをインターネットのタイムサーバーのクロックに同期させるため広く使用されています。 このプロトコルでは、通常 LAN でミリ秒未満の精度が、WAN で最大数ミリ秒の精度が提供されます。 一般的な NTP コンフィグレーションでは、複数の冗長サーバーおよび各種のネットワークパスを使用して、高い精度と信頼性を実現しています。

NTP サブネットは階層レベルにより動作し、各レベルには層と呼ばれる番号が割り付けられます。 第1層(プライマリ)サーバーは最下層レベルにあり、国の時間サービスに直接同期されます。 第2層(セカンダリ)サーバーはその上のレベルにあり、第1層サーバーに同期され、以下同様になります。

7.4 リファレンス情報

時間同期パラメータ

[Properties]ウィンドウで、[Time synchronization]コンフィグレーションエントリを選択します。 TIA ポータルに、[時間同期コンフィグレーション]ダイアログが表示されます。



注記

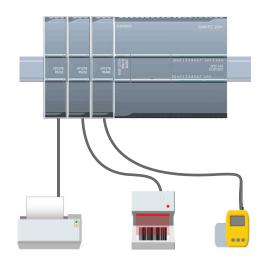
プロジェクトをダウンロードすると、すべての IP アドレスが設定されます。

以下の表は、時間同期のパラメータを定義しています。

パラメータ	定義
Network Time Protocol (NTP) サーバーを使用して時間同期 を有効にする	このチェックボックスをオンにして、NTP サーバーを使用する時間同期を有効にします。
サーバー1	ネットワークタイムサーバー1 に割り付けられた IP アドレス
サーバー2	ネットワークタイムサーバー2 に割り付けられた IP ア ドレス
サーバー3	ネットワークタイムサーバー3 に割り付けられた IP ア ドレス
サーバー4	ネットワークタイムサーバー4 に割り付けられた IP ア ドレス
時間同期間隔	時間間隔値(秒)

ポイントツーポイント(PtP)通信

CPU は、文字ベースのシリアル通信のポイントツーポイントプロトコル(PtP)をサポートしています。シリアル通信では、ユーザーアプリケーションで好みのプロトコルを完全に定義、実装します。 PtP では最大限の自由と柔軟性が得られますが、ユーザープログラム側で多数の実装が必要になります。



PtP により幅広い可能性が提供されます。

- プリンタなどの外部装置に情報を直接送 信する機能
- バーコードリーダー、RFID リーダー、サードパーティ製カメラ、表示システム、 その他の各種装置など他の装置から情報 を読み取る機能
- GPS デバイス、サードパーティ製カメラや表示システム、無線モデムなど、その他の装置とデータの送受信を行って情報を交換する機能

PtP 通信は、標準の UART を使用して各種のボーレートとパリティオプションをサポートするシリアル通信です。 RS232 または RS485 通信モジュール(CM)が、PtP 通信の実行のための電気的インターフェースとなります。

STEP 7 Basic は、ユーザーアプリケーションのプログラミングで使用できる命令のライブラリを提供します。 これらのライブラリは、以下のプロトコルに PtP 通信ファンクションを提供します。

- USS ドライブプロトコル
- Modbus RTU マスタプロトコル
- Modbus RTU スレーブプロトコル

8.1 RS232 および RS485 通信モジュールの使用

2つの通信モジュール(CM)が、PtP通信のインターフェースになります。 CM 1241 RS485 (ページ 441)およびCM 1241 RS232 (ページ 442)です。 任意のタイプのCMを最大 3 つまで接続できます。 CPUまたは他のCMの左側にCMを取り付けます。モジュールの取り付けと取り外しの詳細な説明については、「取り付け」の章 (ページ 34)を参照してください。

RS232 および RS485 通信モジュールには以下の特徴があります。

- 隔離ポート
- ポイントツーポイントプロトコルをサポート
- 拡張命令とライブラリファンクションによって設定およびプログラムされる
- LED によって送受信動作を表示する
- 診断 LED を表示する
- CPU から電源を供給されるため、外部電源接続が不要。

「通信モジュールの技術仕様」 (ページ 441)を参照してください。

8.2 通信ポートのコンフィグレーション

通信モジュールは次の2つの方法で設定できます。

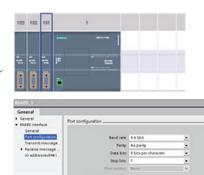
- STEP 7 Basic のデバイスコンフィグレーションを使用してポートパラメータ(ボーおよびパリティ)、送信パラメータと受信パラメータを設定する。 デバイスコンフィグレーション設定は、CPU に永続的に保存されます。 これらの設定は、パワーサイクル後に RUN から STOP モードへ移行した後、適用されます。
- PORT_CFG、SEND_CFG、RCV_CFG 命令を使用して、パラメータを設定します。 命令で設定されたポート設定は、CPU が RUN モードにある間有効です。 ポート設 定は、STOP への移行後またはパワーサイクル後にデバイスコンフィグレーション 設定に戻ります。

ハードウェアデバイスのコンフィグレーション (ページ 91)後に、ラック内のいずれかのCMを選択して、通信インターフェースのパラメータを設定します。

検査ウィンドウの[Properties]タブに、選択された CM のパラメータが表示されます。[Port configuration]を選択して、次のパラメータを編集します。

- ボーレート
- パリティ
- ストップビット数
- フロー制御(RS232 のみ)
- 待機時間

フロー制御以外のポートコンフィグレーションパラメータは、RS232 と RS485 通信モジュールのコンフィグレーションで同じです。 パラメータ値は異なる場合があります。



ポートのコンフィグレーション(または既存のコンフィグレーションの変更)は、ユーザープログラムから PORT_CFG (ページ 348)命令を使用して設定できます。

注記

ユーザープログラムで PORT_CFG 命令を使用して設定したパラメータ値は、STEP 7 Basic から設定したコンフィグレーションを上書きします。 停電時、S7-1200 は PORT_CFG 命令から設定されたパラメータを保持しないことに注意してください。

8.2 通信ポートのコンフィグレーション

ボーレート:ボーレートのデフォルト値は 9.6kbit/秒です。 以下の値を選択できます。

300 ボー 2.4 kbit 19.2 kbit 76.8 kbit 600 ボー 4.8 kbit 28.4 kbit 115.2 kbit 1.2 kbit 57.6 kbit

パリティ: パリティのデフォルト値はパリティなしです。 以下の値を選択できます。

- パリティなし
- 偶数
- 奇数
- マーク(パリティビットは常に1に設定されます)
- スペース(パリティビットは常に0に設定されます)

ストップビット数: ストップビット数は1または2になります。 デフォルトは1です。

フロー制御: RS232 通信モジュールの場合、「フロー制御の管理」 (ページ 335)セクションで説明しているとおり、ハードウェアまたはソフトウェアフロー制御を選択できます。 ハードウェアフロー制御を選択すると、RTS信号を常にオンにするか、RTSを切り替えるかを選択できます。 ソフトウェアフロー制御を選択すると、ASCII文字をXON文字およびXOFF文字に定義できます。

RS485 通信モジュールは、フロー制御をサポートしていません。

待機時間: 待機時間は、通信モジュールが、フロー制御のタイプに応じて、RTS のアサート後に CTS を受信するか、または XOFF の受信後 XON を受信するまでに待つ時間を指定します。 通信モジュールが期待する CTS または XON を受信する前に待機時間が終了すると、通信モジュールは送信操作を中断し、ユーザープログラムにエラーを返します。 待機時間はミリ秒単位で指定します。 範囲は 0~65535 ミリ秒です。

8.3 フロー制御の管理

フロー制御は、データが失われないように、データ転送での送受信のバランスを取るメカニズムのことです。 フロー制御により、受信側のデバイスが処理できる以上の情報を送信側デバイスが送信しないようにします。 フロー制御は、ハードウェアまたはソフトウェアによって実現できます。 RS232 CMは、ハードウェアおよびソフトウェア両方のフロー制御をサポートしています。 RS485 CMは、フロー制御をサポートしていません。 ポートのコンフィグレーション (ページ 333)時、またはPORT_CFG命令を使用して、フロー制御のタイプを指定します。

ハードウェアフロー制御は、送信要求(RTS)および送信可(CTS)通信信号によって動作します。 RS232 CM で、RTS 信号は 7番ピンから出力され、CTS 信号は 8番ピンで受信されます。 CM 1241 は、RTS を出力としてアサートし、CTS を入力としてモニタする DTE (Data Terminal Equipment)デバイスです。

ハードウェアフロー制御: RTS 切り替え

RS232 CMにRTS 切り替えハードウェアフロー制御を有効にすると、モジュールはRTS 信号をアクティブに設定して、データを送信します。 この制御は CTS 信号をモニタして、受信デバイスがデータを受け取れるかどうかを判断します。 CTS 信号が有効の場合、モジュールは CTS 信号がアクティブになっている限りデータを送信できます。CTS 信号が無効になると、送信は停止しなければなりません。

送信は、CTS信号がアクティブになると再開します。 CTS信号が設定された待機時間内にアクティブにならないと、モジュールは送信を中断し、ユーザープログラムにエラーを返します。 待機時間はポートコンフィグレーション (ページ 333)で指定します。 RTS 切り替えフロー制御は、送信が有効になるため信号を必要とするデバイスに便利です。 例えば、ラジオモデムは RTS を「キー」信号として、ラジオトランスミッターを動作させます。 RTS 切り替えフロー制御は、標準の電話モデムでは機能しません。電話モデムの場合は、[RTS always on]を選択してください。

ハードウェアフロー制御: RTS が常にオン

RTS が常にオンのノードで、CM 1241 はデフォルトで RTS を有効に設定します。 電話モデムのようなデバイスは CM から RTS 信号をモニタして、この信号を送信可として利用します。 このモデムは、RTS が有効な場合にのみ CM に送信します。つまり、電話モデムが有効な CTS を確認した場合です。RTS が無効の場合、電話モジュールは CM に送信しません。

モデムがいつでも CM にデータを送信できるようにするには、[RTS always on]ハードウェアフロー制御を設定します。 すると、CM は RTS 信号を常に有効に設定します。 CM は、このモジュールが文字を受け取れない場合でも、RTS を無効に設定しません。 送信側のデバイスは、CM の受信バッファがオーバーランしないようにしなければなりません。

8.3 フロー制御の管理

データ端子ブロックレディ(DTR)およびデータセットレディ(DSR)信号の使用

CM はいずれのハードウェアフロー制御についても DTR を有効に設定します。 このモジュールは、DSR 信号が有効になる場合だけ送信します。 DSR の状態は、送信操作の開始時にのみ評価されます。 送信開始後に DSR が無効になった場合、送信は停止しません。

ソフトウェアフロー制御

ソフトウェアフロー制御は、フロー制御を行うために特殊文字を使用します。 これらの文字は XON と XOFF を示す ASCII 文字です。

XOFF は送信を停止することを示します。 XON は送信を再開することを示します。

送信側のデバイスが受信側のデバイスからXOFF文字を受信すると、送信は停止します。 送信は、送信側のデバイスがXON文字を受信すると再開します。 送信側デバイスが、 ポートコンフィグレーション (ページ 333)で指定された待機時間内にXON文字を受信し ないと、このCMは送信を中断し、ユーザープログラムにエラーを返します。

ソフトウェアフロー制御には全二重通信が必要です。受信側のパートナーが、送信中に 送信側のパートナーに XOFF を送信する必要があるからです。 ソフトウェアフロー制御は、ASCII 文字を含むメッセージでのみ使用可能です。 バイナリプロトコルでは、ソフトウェアフロー制御は使用できません。

8.4 送信パラメータと受信パラメータのコンフィグレーション

PLC が PtP 通信を開始できるようにするには、メッセージの送信パラメータと受信パラメータを設定する必要があります。 これらのパラメータにより、ターゲット装置とのメッセージの送受信での通信動作が決まります。

送信パラメータのコンフィグレーション



CM のコンフィグレーション 中、選択した CM について [Transmit message configuration]プロパティを指定 して、通信インターフェースの データ送信方法を設定します。

SEND_CFG (ページ 351)命令を使用して、ユーザープログラムから送信メッセージのパラメータをダイナミックに設定または変更することもできます。

注記

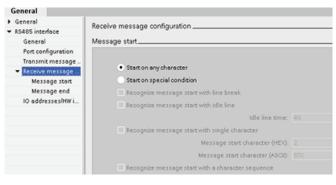
ユーザープログラムの SEND_CFG 命令から設定したパラメータ値は、ポートコンフィグレーションの設定を上書きします。 停電時、CPU は PORT_CFG 命令から設定されたパラメータを保持しないことに注意してください。

パラメータ	定義
RTS オン遅延	送信が開始される前にRTSの有効化を待機する時間の量を指定します。 範囲は 0~65535 msで、デフォルト値は 0 です。このパラメータは、ポートコンフィグレーション (ページ 333)でハードウェアフロー制御が指定されている場合にのみ有効です。CTSは、RTSオン遅延時間が終了した後に評価されます。
	このパラメータは RS232 モジュールにのみ適用されます。
RTS オフ遅延	送信の完了後RTSを無効化する前に待機する時間の量を指定します。 範囲は 0~65535 msで、デフォルト値は 0 です。このパラメータは、ポートコンフィグレーション (ページ 333)でハードウェアフロー制御が指定されている場合にのみ有効です。このパラメータは RS232 モジュールにのみ適用されます。

8.4 送信パラメータと受信パラメータのコンフィグレーション

パラメータ	定義
メッセージ開始時に ブレーク信号を送信 1 つのブレーク信号	各メッセージの開始時に、RTS オン遅延(設定されている場合)が終了し、CTS が有効な場合、ブレーク信号が送信されることを指定します。
内のビット時間数	ブレークの間、回線が間隔条件で保持される 1 つのブレークを 構成するビット時間数を指定します。 デフォルトは 12 で最大 値は 65535 です(8 秒が上限)。
ブレーク信号後アイ ドルラインを送信 ブレーク後のアイド ルライン	メッセージ開始時のブレーク信号後、アイドルラインが送信されることを指定します。 [Idle line after a break]パラメータは、マーキング条件でラインが保持されるアイドルラインを構成するビット時間数を指定します。 デフォルトは 12 で最大値は65535 です(8 秒が上限)。

受信パラメータのコンフィグレーション



デバイスコンフィグレーションから、通信インターフェースがデータを受信する方法、およびメッセージの開始と終了の両方を認識する方法を設定します。これらのパラメータは、選択した CM の受信メッセージのコンフィグレーションで指定します。

RCV_CFG (ページ 353)命令を使用して、ユーザープログラムから受信メッセージのパラメータを動的に設定または変更することもできます。

注記

ユーザープログラムの RCV_CFG 命令から設定したパラメータ値は、ポートコンフィグレーションの設定を上書きします。 停電時、CPU は RCV_CFG 命令から設定されたパラメータを保持しないことに注意してください。

詳細については、「RCV_CFG命令」を参照してください。

メッセージ開始パラメータ

通信モジュールが、メッセージの開始を認識する方法を指定できます。 開始文字とメッセージを構成する文字は、設定された終了条件が満たされるまで、受信バッファに入ります。

複数の開始条件を指定できます。 すべての開始条件は、メッセージが開始されたと見なされる前に満たされる必要があります。 たとえば、アイドルライン時間と特定の開始文字を設定すると、CM は最初にアイドルライン要件が満たされてから、指定された開始文字を確認します。 開始文字として指定されていないその他の文字が受信されると、CM はもう一度アイドルライン時間を確認して、メッセージ検索開始を再開します。

- アイドルライン
- 改行
- 文字または文字のシーケンス

開始条件の確認順序は次のとおりです。

複数の開始条件の確認中、一方の条件が満たされない場合、CM は最初に必要な条件で確認を再開します。

パラメータ	定義
開始文字	開始文字条件は、特定の文字が正しく受信されるとメッセージを開始することを指定します。 この文字は、メッセージ内の最初の文字になります。 この特定の文字より前に受信された文字は、すべて破棄されます。
任意の文字で開始	任意の文字で開始条件は、任意の文字が正しく受信されるとメッセージを開始することを指定します。 この文字は、メッセージ内の最初の文字になります。
改行文字	改行文字条件は、メッセージ受信操作は改行文字が受信された 後に開始することを指定します。
アイドルライン	アイドルライン条件は、受信ラインが指定したビット時間数の間アイドルまたは無活動であった場合に、メッセージ受信を開始することを指定します。 この条件が発生すると、メッセージ受信を開始します。
特殊な条件: 1 文字でメッセージ 開始を認識	特定の1文字がメッセージ開始を示すことを指定します。 デフォルトは STX です。

8.4 送信パラメータと受信パラメータのコンフィグレーション

パラメータ	定義
特殊な条件: 文字シーケンスに よってメッセージ 開始を認識	特定の文字シーケンスがメッセージ開始を示すことを指定します。各シーケンスについて、最大 5 文字を指定できます。各文字の位置について、1 文字の特定の 16 進文字を指定するか、またはシーケンスマッチングでその文字を無視することを指定できます。 受信シーケンスは、いずれかの開始条件が満たされるまで、指定された複数の開始条件について評価されます。いずれかの開始条件が満たされると、終了条件の評価が開始します。必要に応じて、最大 5 文字の特定文字のシーケンスを有効または無効にすることを設定できます。開始条件は、設定された文
	字シーケンスのいずれかが発生すると満たされます。
サンプルコンフィ グレーション	✓ Recognice message start with a character sequence Number of character sequences to define: 2 5-character message start sequences Message start sequence 1 Check this character: ✓ Character value (HEQ): 6A D O O O O O O O O O O O O
	Message start sequence 2 Check this character: Character value (HEX): 0 6A 6A 0 0 Character value (ASCII): ANY 1 NUL ANY ANY
	このコンフィグレーションで、開始条件は以下のいずれかのパ ターンが発生すると満たされます。
	 最初の文字が 0x6A で、かつ5番目の文字が 0x1C である5 文字のシーケンスが受信されたとき。2番目、3番目、4番 目の文字は、このコンフィグレーションでは何の文字であってもかまいません。5番目の文字が受信されると、終了条件の評価が開始します。 前に任意の1文字があり、その後に2つの 0x6A 文字が連続して受信されるとき。この場合、終了条件の評価は、2番目
	の 0x6A が受信されるとす。この場合、終了来件の評価は、2番日の 0x6A が受信されると開始します(3 文字)。 最初の 0x6A 文字の前の文字は、開始条件に含まれます。 この開始条件を満たすサンプルシーケンスを以下に示します。
	< <任意の文字> 6A 6A
	• 6A 12 14 18 1C
	• 6A 44 A5 D2 1C

メッセージ終了パラメータ

通信インターフェースが、メッセージの終了を認識する方法を設定できます。 複数の メッセージ終了条件を設定可能です。 設定された条件のいずれかが発生すると、メッ セージは終了します。

複数の終了条件を同時に指定することができます。 メッセージは、いずれかの終了条件が満たされると終了します。 例えば、300 ミリ秒のメッセージタイムアウトの終了、40 ビット時間の文字間タイムアウト、最大長 50 バイトでの終了条件を指定できます。 メッセージは、受信に 300 ミリ秒より長い時間がかかるメッセージの場合、または任意の 2 文字間のギャップが 40 ビット時間を超える場合、または 50 バイトを受信すると終了します。

パラメータ	定義
メッセージタイムア ウトによりメッセー ジ終了を認識	メッセージ終了は、メッセージ終了を待つ設定時間数が終了すると発生します。 メッセージタイムアウト時間は、メッセージ 開始条件に従って最初の文字が受信されると開始します。 デフォルトは 200 ms で、範囲は 0~65535 ms です。
応答タイムアウトに よってメッセージ終 了を認識	メッセージの終了は、有効なシーケンスが受信される前に、応答を待つ設定時間数が終了すると発生します。 応答タイムアウト時間は、送信が終了すると開始します。 デフォルトの応答タイムアウトは 200 ms で、範囲は 0~65535 ms です。メッセージの実際の終了を示すには、別の終了条件を設定する必要があります。
文字間ギャップによってメッセージ終了 を認識	メッセージの終了は、連続する文字間の最大設定時間数が終了すると発生します。 文字間ギャップのデフォルト値は 12 ビット時間で、最大は 65535 ビット時間で 8 秒です。
最大長によってメッ セージ終了を認識	メッセージの終了は、設定された最大文字数が受信されると発生します。 デフォルトは 0 バイトで、最大は 1024 バイトです。
メッセージからメッセージ長を読み取り	メッセージ自体が、メッセージの長さを指定します。 メッセージの終了は、指定された長さのメッセージが受信されると発生します。 メッセージ長の指定と解釈方法を以下に説明します。
1 文字でメッセージ 終了を認識	メッセージ終了は、指定された 1 文字が受信されると発生します。

8.4 送信パラメータと受信パラメータのコンフィグレーション

パラメータ	定義
文字シーケンスによ りメッセージ終了を 認識	メッセージ終了は、指定された文字シーケンスが受信されると 発生します。 1 つのシーケンスについて、最大 5 文字を指定できます。 各文字の位置について、1 文字の特定の 16 進文字を指定するか、またはシーケンスマッチングでその文字を無視することを指定できます。 無視される文字である先頭文字は、終了条件の一部ではありません。 無視される文字である終了文字は、終了条件の一部ではありません。
サンプルコンフィグレーション	▼Recognize message end with a character sequence S-character message end sequence Check this character Check this character (MEX): 7A

メッセージ内のメッセージ長の指定

メッセージ長がメッセージに含まれるという特殊な条件を選択した場合、メッセージ長に関する情報を定義する3つのパラメータを指定する必要があります。

実際のメッセージ構造は、使用するプロトコルによって異なります。 3 つのパラメータは次のとおりです。

- n: 長さ指定子を開始するメッセージ内の文字の位置(1 から開始)
- 長さのサイズ: 長さ指定子のバイト数(1、2、または 4)
- 長さ m: 長さのカウントに含まれない、長さ指定子に続く文字数



これらのフィールドは、デバイスプロ パティの受信メッセージのコンフィグ レーションに表示されます。

8.4 送信パラメータと受信パラメータのコンフィグレーション

例1:以下のプロトコルに従って構成されたメッセージを考えます。

STX	Len		長さによってカウントされる文字 3~14										
	(n)	ADR	PKE		INDEX		PWD		STW		HSW		всс
1	2	3	4	5	6	7	8	9	10	11	12	13	14
STX	0x0 C	xx	XXXX		xxxx		XXXX		XXXX		XXXX		xx

この受信メッセージの長さパラメータを以下のように設定します。

- n = 2 (メッセージの長さは2バイト目で開始します。)
- 長さのサイズ = 1 (メッセージの長さは 1 バイトで定義されます。)
- 長さ m = 0 (長さのカウントに含まれない長さ指定子に続く追加文字はありません。 12 文字が長さ指定子に続きます。)

この例で、文字 3~14 (3 と 14 も含む)は、Len (n)でカウントされる文字です。

例 2: 以下のプロトコルに従って構成された別のメッセージを考えます。

SD1	Len	Len	SD2	長さり	こよって	カウン	トされ	る文字:	FC	ED	
	(n)	(n)		DA	SA	FA	データ単位=3 バイ				
							٢				
1	2	3	4	5	6	7	8	9	10	11	12
XX	0x06	0x06	xx	xx	xx	xx	xx	xx	xx	xx	xx

この受信メッセージの長さパラメータを以下のように設定します。

- n=3(メッセージの長さは3バイト目で開始します。)
- 長さのサイズ = 1 (メッセージの長さは1バイトで定義されます。)
- 長さ m = 3 (長さのカウントに含まれない長さ指定子に続く 3 文字があります。 この例のプロトコルでは、文字 SD2、FCS、ED は長さのカウントに入りません。 他の 6 文字は長さのカウントに数えられるため、長さ指定子に続く文字数合計は 9 です。)

この例で、文字 5~10 (5 と 10 も含む)は、Len (n)でカウントされる文字です。

8.5 ポイントツーポイント通信のプログラミング

STEP 7 Basic では、ユーザープログラムで設計、指定されたポイントツーポイント通信を実行可能にする拡張命令を提供しています。 これらの命令は次の 2 つのカテゴリに分けられます。

- コンフィグレーションに関する注意事項
- 通信命令

コンフィグレーションに関する注意事項

ユーザープログラムでポイントツーポイント通信を実行可能にするには、データの送受信に使用する通信インターフェースポートとパラメータを最初に設定する必要があります。

各通信モジュールのポートとメッセージのコンフィグレーション • PORT_CFG は、ユーザープログラム内のデバイスコンフィグレーションまたは • SEND_CFG その命令によって実行することができます。 • RCV CFG

通信命令

ポイントツーポイント通信命令により、ユーザープログラムで通信モジュール間のメッセージの送受信を行うことができます。 これらの命令によるデータ転送の詳細については、データの一貫性 (ページ 114)に関するセクションを参照してください。

すべてのポイントツーポイントファンクションは非同期で動作します。 ユーザープログラムはポーリングアーキテクチャを使用して、送受信ステータスを判別できます。 SEND_PTP と RCV_PTP は同時に実行できます。 通信モジュールは、最大バッファサイズ 1024 バイトの送受信メッセージを必要に応じてバッファします。

通信モジュールは、実際のポイントツーポイントデバイスからメッ • SEND_PTP セージを送受信します。 特定の通信ポートから送受信されたメッ • RCV_PTP セージプロトコルがバッファの中にあります。

追加の命令により、受信バッファのリセット機能、特定の RS232• RCV_RST信号の受信と設定機能が提供されます。• SGN_GET

SGN_SET

8.5.1 ポーリングアーキテクチャ

S7-1200 ポイントツーポイント命令は、周期的/定期的に呼び出して受信メッセージを確認する必要があります。 ユーザープログラムは、送信のポーリングにより転送が完了すると通知を受けます。

ポーリングアーキテクチャ:マスタ

マスタの一般的なシーケンスは次のとおりです。

- 1. SEND PTP 命令が通信モジュールへの送信を開始します。
- 2. SEND_PTP 命令が以降のスキャンに実行され、送信完了ステータスのポーリングを行います。
- 3. SEND_PTP 命令が送信完了を示すと、ユーザーコードは応答を受信する準備を行う ことができます。
- 4. RCV_PTP 命令が繰り返し実行され、応答が確認されます。 通信モジュール(CM)が 応答メッセージを収集すると、RCV_PTP 命令はこの応答を CPU にコピーし、新規 データが受信されたことを示します。
- 5. ユーザープログラムはこの応答を処理できます。
- 6. ステップ 1 に戻りこの手順を繰り返します。

ポーリングアーキテクチャ: スレーブ

スレーブの一般的なシーケンスは次のとおりです。

- 1. ユーザープログラムは、スキャンごとに RCV PTP 命令を実行する必要があります。
- 2. 通信モジュール(CM)が要求を受信すると、RCV_PTP 命令は新規データの準備ができたことを示し、要求は CPU にコピーされます。
- 3. ユーザープログラムは、この要求に応え、応答を生成する必要があります。
- 4. SEND_PTP 命令を使用して、応答をマスタに返送します。
- 5. SEND_PTP を繰り返し実行して、送信が確実に発生するようにします。
- 6. ステップ 1 に戻りこの手順を繰り返します。

スレーブは RCV_PTP を頻繁に呼び出して、応答の待機中にマスタがタイムアウトする前に、マスタからの送信を受信できるようにする役割を担います。 このタスクを実行するために、ユーザープログラムは、タイムアウト時間が終了する前にマスタからの送信を受信する十分なサイクルタイムがある場合、サイクリック OB から RCV_PTP を呼び出すことができます。 マスタのタイムアウト時間内に 2 つの実行を行うように OB のサイクルタイムを設定した場合、ユーザープログラムは何も失うことなく送信を受信するはずです。

8.6 ポイントツーポイント命令

8.6.1 ポイントツーポイント命令の一般的なパラメータ

通信モジュール LED の動作

通信モジュール(CM)には3つのLEDインジケータがあります。

- 診断 LED: この LED は CPU によってアドレス指定されるまで赤く点滅します。 CPU に電源が入ると、モジュールを確認して CM モジュールをアドレス指定します。 診断 LED は緑に点滅し始めます。 これは、CPU が CM をアドレス指定したがコンフィグレーションを提供していないことを示します。コンフィグレーションは、プログラムが CPU にダウンロードされるとモジュールにダウンロードされます。 CPU にダウンロードされると、通信モジュールの診断 LED は緑に点灯します。
- 送信 LED: この LED は受信 LED の上にあります。 送信 LED は、データが通信ポートから送出されているときに点灯します。
- 受信 LED: この LED は、データが通信ポートによって受信されているときに点灯します。

ビット時間解像度

設定済みのボーレートによって、あるビット時間数単位で指定された複数のパラメータ。このパラメータをビット時間単位で指定すると、パラメータがボーレートに依存しなくなります。 すべてのパラメータは、ビット時間数単位で最大数 65535 に指定できます。ただし、S7-1200 で測定可能な最大時間数は8秒です。

REQ 入力パラメータ

多くのポイントツーポイント(PtP)命令では、LOW から HIGH への移行操作を開始する REQ 入力を使用します。 REQ 入力は、1 つの命令実行では HIGH (TRUE)でなければ なりませんが、REQ 入力は必要なだけ TRUE のままにすることができます。 この命令は、REQ 入力が FALSE で呼び出されるまで他の操作を開始しないので、命令は REQ 入力の履歴状態をリセットできます。 これは、命令が LOW から HIGH への移行を検出し、次の操作を開始できるようにするために必要です。

PtP 命令を指定すると、インスタンス DB を指定するように指示されます。 各 PtP 命令について一意の DB を使用します。 つまり、あるポートに対するすべての SEND_PTP 命令には同じインスタンス DB が必要ですが、SEND_PTP と RCV_PTP には異なるインスタンス DB が必要になります。 これにより、REQ などの入力が各命令で正しく処理されます。

PORT 入力パラメータ

PORT 入力に関するドロップダウンメニューから、この命令のインスタンスが操作する CM のポート識別子を選択します。 この番号は CM のコンフィグレーション情報で「ハードウェア識別子」とも呼ばれます。

NDR、DONE、ERROR、STATUS 出力パラメータ

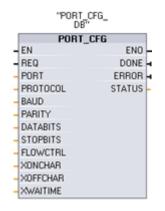
- 出力 **DONE** は、要求された操作がエラーなしで完了したことを示します。 この出力は **1** つのスキャンに対してセットされます。
- 出力 NDR (New Data Ready)は、要求された操作がエラーなしで完了し、新しいデータが受信されたことを示します。 この出力は 1 つのスキャンに対してセットされます。
- 出力 ERROR は、要求された操作がエラーで終了したことを示します。 この出力は 1 つのスキャンに対してセットされます。
- 出力 STATUS は、エラーまたは中間ステータス結果をレポートするために使用されます。
 - DONE または NDR ビットがセットされると、STATUS は 0 または情報コードに設定されます。
 - ERROR ビットがセットされると、STATUS はエラーコードに設定されます。
 - 上記のいずれのビットも設定されない場合、命令はビジーステータスなどの、ファンクションの現在の状態を記述する結果を返します。

8.6 ポイントツーポイント命令

一般的な条件コード

STATUS (W#16#)	説明
0000	エラーは発生していません。
8x3A	パラメータ x に不正なポインタがあります。
8070	すべての内部インスタンスメモリが使用中です。
8080	ポート番号が不正です。
8081	タイムアウト、モジュールエラー、または他の内部エラーです。
8082	パラメータ設定は、パラメータ設定がバックグラウンドで進行中のため失敗しました。
8083	バッファオーバーフローです。
	CMが、長さパラメータが許可するよりも長い受信メッセージを返しました。
8090	メッセージ長、サブモジュールが間違っているか、不正なメッセージ です。
8091	パラメータ設定メッセージのバージョンが間違っています。
8092	パラメータ設定メッセージのレコード長が間違っています。

8.6.2 PORT_CFG命令



PORT_CFG (Port Configuration)では、ボーレートなどのポートパラメータをユーザープログラムから変更できます。ポートの初期スタティックコンフィグレーションをデバイスのコンフィグレーションプロパティに設定するか、デフォルト値を使用できます。 PORT_CFG 命令をユーザープログラムで実行して、コンフィグレーションを変更できます。 PORT_CFG コンフィグレーションの変更は、CPU に永続的には保存されません。 デバイスコンフィグレーションで設定されたパラメータは、CPU が RUN から STOP モードに移行し、1 つの電源サイクルが終わった後で復元されます。詳細については、「通信ポートの設定」 (ページ 333)と「フロー制御の管理」 (ページ 335)を参照してください。

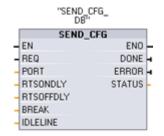
パラメータ	パラメ	データタ	説明
		イプ	
DE0	イプ	D001	
REQ	IN	BOOL	この入力信号の立ち上がりエッジのコンフィグ
			レーション変更を有効にします。
PORT	IN	PORT	通信ポート識別子:
			この論理アドレスは定数で、デフォルトタグテ
			ーブルの[Constants]タブで参照できます。
PROTOCOL	IN	UInt	0 - ポイントツーポイント通信プロトコル
			1n - 特定プロトコルの将来の定義
BAUD	IN	UInt	ポートボーレート:
			1 - 300 ボー
			2 - 600 ボー
			3 - 1200 ボー
			4 - 2400 ボー
			5 - 4800 ボー
			6 - 9600 ボー
			7 - 19200 ボー
			8 - 38400 ボー
			9 - 57600 ボー
			10 - 76800 ボー
			11 - 115200 ボー
PARITY	IN	UInt	ポートパリティ:
			1 - パリティなし
			2 - 偶数パリティ
			3 - 奇数パリティ
			4-マークパリティ
			5 - スペースパリティ
DATABITS	IN	UInt	キャラクタ当たりビット:
			1-8データビット
			2-7データビット
STOPBITS	IN	UInt	ストップビット:
			1-1ストップビット
			2-2ストップビット

8.6 ポイントツーポイント命令

パラメータ	パラメ	データタ	説明
	ータタ	イプ	
	イプ		
FLOWCTRL	IN	UInt	フロー制御:
			1-フロー制御なし
			2 - XON/XOFF
			3 - ハードウェア RTS 常に ON
			4 - ハードウェア RTS 切り替え
XONCHAR	IN	CHAR	XON 文字として使用する文字を指定します。 通
			常これは DC1 文字(11H)です。 このパラメータ
			はフロー制御が有効な場合のみ評価されます。
XOFFCHAR	IN	CHAR	XOFF 文字として使用する文字を指定します。
			通常これは DC3 文字(13H)です。 このパラメー
			タはフロー制御が有効な場合のみ評価されま
			す。
XWAITIME	IN	UInt	XOFF 文字の受信後に XON 文字を待つ時間、ま
			たは RTC を有効にした後 CTS 信号を待つ時間
			を指定します(0~65535 ms)。 このパラメータは
			フロー制御が有効な場合のみ評価されます。
DONE	OUT	BOOL	1 スキャンについて、最後の要求がエラーなしで
			完了した後、TRUE になります。
ERROR	OUT	Bool	1 スキャンについて、最後の要求がエラーで終了
			した後、TRUE になります。
STATUS	OUT	Word	実行条件コード

STATUS (W#16#)	説明
80A0	特定のプロトコルが存在しません。
80A1	特定のボーレートが存在しません。
80A2	特定のパリティオプションが存在しません。
80A3	特定のデータビット数が存在しません。
80A4	特定のストップビット数が存在しません。
80A5	特定のフロー制御タイプが存在しません。
80A6	待ち時間0とフロー制御が有効です。
80A7	XON と XOFF の値が不正です。

8.6.3 SEND_CFG命令



SEND_CFG (Send Configuration)を使って、ポイントツーポイント通信ポートのシリアル転送パラメータのダイナミックコンフィグレーションが可能です。 通信モジュール(CM)のキューに登録されたメッセージは、SEND_CFG が実行されるとすべて破棄されます。

ポートの初期スタティックコンフィグレーションをデバイスのコンフィグレーションプロパティに設定するか、デフォルト値を使用できます。 SEND_CFG命令をユーザープログラムで実行して、コンフィグレーションを変更できます。 SEND_CFGコンフィグレーションの変更は、PLCに永続的には保存されません。 デバイスコンフィグレーションで設定されたパラメータは、CPUがRUNからSTOPモードに移行し、1 つの電源サイクルが終わった後で復元されます。 「送受信パラメータの設定」 (ページ 337)を参照してください。

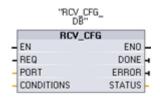
パラメータ	パラメー タタイプ	データタ イプ	説明
REQ	IN	BOOL	この入力信号の立ち上がりエッジのコンフィグレーション変更を有効にします。
PORT	IN	PORT	通信ポート識別子: この論理アドレスは定数で、デフォルトタグテーブルの[Constants]タブで参照できます。
RTSONDLY	IN	UInt	RTS を有効にした後、Tx データ転送が発生する前に待機するミリ秒数。 このパラメータは ハードウェアフロー制御が有効な場合のみ有効です。 0 - 65535 ms. 0 はこの機能を無効にします。
RTSOFFDLY	IN	UInt	RTS を無効にする前、Tx データ転送が発生する後に待機するミリ秒数。 このパラメータは ハードウェアフロー制御が有効な場合のみ有効です。 0 - 65535 ms. 0 はこの機能を無効にします。

8.6 ポイントツーポイント命令

パラメータ	パラメー タタイプ	データタ イプ	説明
BREAK	IN	UInt	このパラメータは、指定したビット時間数の間、各メッセージの開始後に改行文字が送信されることを示します。 最大値は 65535 ビット時間です。 0 はこの機能を無効にします。 最小8秒
IDLELINE	IN	UInt	このパラメータは、指定したビット時間数の間、各メッセージの開始後にラインがアイドルのままになることを示します。 最大値は65535 ビット時間です。 0 はこの機能を無効にします。 最小8秒
DONE	OUT	BOOL	1 スキャンについて、最後の要求がエラーなしで完了した後、TRUE になります。
ERROR	OUT	BOOL	1 スキャンについて、最後の要求がエラーで終了した後、TRUE になります。
STATUS	OUT	Word	実行条件コード

STATUS (W#16#	説明
80B0	送信割り込みのコンフィグレーションはできません
80B1	切断時間が許容値(2500 ビット時間)よりも大きくなっています
80B2	アイドル時間が許容値(2500 ビット時間)よりも大きくなっています

8.6.4 RCV_CFG命令



RCV_CFG (Receive Configuration)は、ポイントツーポイント通信ポートのシリアル受信パラメータのダイナミックコンフィグレーションを実行します。 この命令は、受信メッセージの開始と終了を知らせる条件を設定します。 CM にキュー登録されたメッセージは、RCV_CFG が実行されるとすべて破棄されます。

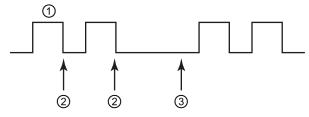
CMポートの初期スタティックコンフィグレーションをデバイスのコンフィグレーションプロパティに設定するか、デフォルト値を使用できます。 RCV_CFG命令をユーザープログラムで実行して、コンフィグレーションを変更できます。 RCV_CFGコンフィグレーションの変更は、PLCに永続的には保存されません。 デバイスコンフィグレーションで設定されたパラメータは、CPUがRUNからSTOPモードに移行し、1 つの電源サイクルが終わった後で復元されます。 詳細については、「受信パラメータの設定」 (ページ 337)を参照してください。

パラメータ	パラメー タタイプ	データタイプ	説明
REQ	IN	BOOL	この入力信号の立ち上がりエッジのコンフィグレーション変更を有効にします。
PORT	IN	PORT	通信ポート識別子: この論理アドレスは定数で、デフォルトタ グテーブルの[Constants]タブで参照できま す。
CONDITION S	IN	CONDITIONS	条件データ構造は、メッセージ条件の開始 と終了を指定します。 これらについて は、以下に説明します。
DONE	OUT	Bool	1 スキャンについて、最後の要求がエラーなしで完了した後、TRUE になります。
ERROR	OUT	Bool	1 スキャンについて、最後の要求がエラー で終了した後、TRUE になります。
STATUS	OUT	Word	実行条件コード

RCV_PTP 命令の開始条件

RCV_PTP 条件は RCV_CFG 命令で指定されたコンフィグレーションを使用して、ポイントツーポイント通信メッセージの開始と終了を指定します。 メッセージの開始は開始条件によって指定されます。 メッセージの開始は、1 つまたは複数の開始条件の組み合わせで指定できます。 複数の開始条件が指定された場合、それらすべての条件が満たされないとメッセージは開始されません。 指定可能な開始条件は次のとおりです。

- 開始文字は、特定の文字が正しく受信されるとメッセージを開始することを指定します。 この文字は、メッセージ内の最初の文字になります。 この特定の文字より前に受信された文字は、すべて破棄されます。
- 任意の文字は、任意の文字が正しく受信されるとメッセージを開始することを指定 します。 この文字は、メッセージ内の最初の文字になります。
- 改行文字は、メッセージ受信操作は改行文字が受信された後に開始することを指定 します。
- アイドルラインは、受信ラインが指定したビット時間数の間アイドル状態または非動作状態であった場合に、メッセージ受信を開始することを指定します。 この条件が発生すると、メッセージ受信を開始します。



- ① 文字
- ② アイドルラインタイマを再起動します
- ③ アイドルラインが検出され、メッセージ受信を開始します
- 可変シーケンス: 開始条件は、文字シーケンスの可変数(最大数 4)に基づいて構築できます。これらは文字の可変数(最大数 5)で構成されます。 シーケンス内での各文字の位置は、特定の文字として選択するか、ワイルドカード文字(任意の文字が一致)として選択することができます。 この開始条件は、文字の異なるシーケンスがメッセージの開始を示す場合に使用できます。

次の16 進コード受信メッセージを検討してください。"「68 10 aa 68 bb 10 aa 16」および設定済みの開始シーケンスを下の表に示します。 開始シーケンスは、最初の68H文字が正しく受信されると評価が開始します。4番目の文字(2番目の68H)が正しく受信されると、開始条件1が満たされます。 開始条件が満たされると、終了条件の評価が開始します。

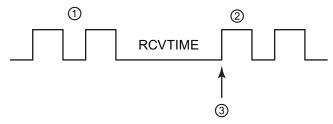
開始シーケンスの処理は、各種パリティエラー、フレーミングエラー、文字間エラーが原因で終了する場合があります。 これらのエラーは、開始条件が満たされていないため、受信メッセージなしという結果になります。

開始条件	最初の文字	最初の文字 +1	最初の文字 +2	最初の文字 +3	最初の文字 +4
1	68 H	xx	xx	68H	xx
2	10H	ааН	xx	xx	xx
3	dcH	ааН	xx	xx	xx
4	e5H	xx	xx	xx	xx

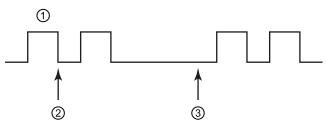
RCV_PTP 命令の終了条件

メッセージの終了は終了条件によって指定されます。 メッセージの終了は、1つまたは 複数の設定済み終了条件の最初の発生によって指定されます。 指定可能なメッセージ 終了条件は次のとおりです。

• 応答タイムアウトは、RCVTIME で指定された時間内に応答文字が正しく受信されるべきであることを指定します。 タイマは送信が正しく完了するとただちに開始し、モジュールは受信操作を開始します。 RCVTIME 時間内に文字が受信されないと、対応する RCV_PTP 命令にエラーが返されます。 応答タイムアウトは特定の終了条件を定義しません。 文字が指定時間内に正しく受信されるべきであることだけを指定します。 応答メッセージの終了条件を定義するには、明確な終了条件を使用する必要があります。

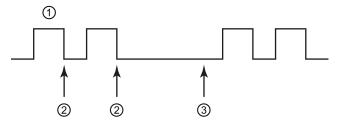


- ① 送信済み文字
- ② 受信済み文字
- ③ 最初の文字がこの時点で正しく受信されている必要があります
- メッセージタイムアウトは、MSGTIME で指定された時間内にメッセージが受信されなければならないことを指定します。 タイマは、指定済みの開始条件が満たされると、ただちに開始します。



- ① 受信済み文字
- ② 開始メッセージ条件が満たされました。 メッセージタイマが開始します
- ③ メッセージタイマが期限切れになって、メッセージが終了します

• 文字間ギャップは、1つの文字の終了(最後のストップビット)から次の文字の終了までの測定時間です。 任意の2文字間の時間が設定済みのビット時間数を超えると、メッセージは終了します。



- ① 受信済み文字
- ② 文字間タイマが再起動します
- ③ 文字間タイマが期限切れになって、メッセージがエラーで終了します
- 最大長: 受信操作は、指定済みの文字数を受信すると停止します。 この条件は、メッセージバッファオーバーランエラーの防止に使用できます。

この終了条件をタイムアウト終了条件と組み合わせ、タイムアウト条件が発生すると、有効な受信済み文字は、最大長に達していなくてもすべて供給されます。 これにより、最大長だけがわかっている場合に、可変長プロトコルをサポートすることができます。

- 組み合わせ条件「N+長さのサイズ+長さ M」。 この終了条件は、長さフィールドを 含む可変サイズメッセージの処理に使用できます。
 - Nは、長さフィールドが開始する位置(メッセージに入る文字数)を指定します (1 をベース)。
 - 長さのサイズは、長さフィールドのサイズを指定します。 有効な値は 1、2、4 バイトです。
 - 長さ M は、メッセージの長さに含められない終了文字(長さフィールドに続く)を 指定します。 この値は、長さフィールドに含められないサイズのチェックサムフィールドの長さを指定するために使用できます。
 - たとえば、あるメッセージフォーマットが、開始文字、アドレス文字、1バイトの長さフィールド、メッセージデータ、チェックサム文字、終了文字で構成されているとします。「Len」で示されるエントリは N パラメータに対応します。 N の値は 3 で、長さバイトがメッセージの 3 バイト目に位置することを示します。長さのサイズの値は 1 で、メッセージの長さの値は、1 バイト内に含まれることを示します。チェックサムフィールドと終了文字フィールドは、「長さ M」パラメータに対応します。「長さ M」の値は 3 で、チェックサムフィールドと終了文字フィールドのバイト数を示します。

8.6 ポイントツーポイント命令

開始文字	アドレス	Len	メッ1	セージ	チェッ	クサムと終	·了文字
		(N)				長さM	
(1)	(2)	(3)		(x))	(+1 x+2 x+	3
xx	xx	xx	xx	xx	xx	xx	xx

• 可変文字: この終了条件は、異なる文字シーケンスに基づいて、受信を終了するために使用します。シーケンスは、異なる数の文字で構成できます(最大 5 文字)。シーケンス内での各文字の位置は、特定の文字として選択するか、ワイルドカード文字(任意の文字が条件を満足する)として選択することができます。 無視するように設定された任意の最初の文字は、メッセージの一部である必要はありません。 無視される任意の最後の文字は、メッセージの一部でなければなりません。

CONDITIONS データタイプ構造体のパラメータパート 1 (開始条件)

パラメータ	パラメー タタイプ		説明
STARTCOND	IN	UInt	開始条件を指定します。
			• 01H - 開始文字
			• 02H - 任意の文字
			• 04H - 改行文字
			• 08H - アイドルライン
			• 10H - シーケンス 1
			• 20H - シーケンス 2
			• 40H - シーケンス 3
			• 80H - シーケンス 4
IDLETIME	IN	UInt	アイドルラインタイムアウトに必要なビット時
			間数。 アイドルライン条件でのみ使用されま
			す。 0 - 65535
STARTCHAR	IN	Byte	開始文字条件で使用される開始文字。

パラメータ	パラメー	データタ	説明
	タタイプ	イプ	
STRSEQ1CTL	IN	Byte	シーケンス 1 は各文字のコントロールを無視/
			比較します。
			これらのコントロールは、開始シーケンスでの
			各文字のイネーブルビットです
			• 01H - 文字 1
			• 02H - 文字 2
			• 04H - 文字 3
			• 08H - 文字 4
			• 10H - 文字 5
			文字に関連付けられたこのビットを無効にする
			と、このシーケンス位置で任意の文字が一致す
			ることになります。
STRSEQ1	IN	Char[5]	シーケンス 1 の開始文字(5 文字)
STRSEQ2CTL	IN	Byte	シーケンス2は各文字のコントロールを無視/
			比較します
STRSEQ2	IN	Char[5]	シーケンス 2 の開始文字(5 文字)
STRSEQ3CTL	IN	Byte	シーケンス3は各文字のコントロールを無視/
			比較します
STRSEQ3	IN	Char[5]	シーケンス 3 の開始文字(5 文字)
STRSEQ4CTL	IN	Byte	シーケンス 4 は各文字のコントロールを無視/
			比較します
STRSEQ4	IN	Char[5]	シーケンス 4 の開始文字(5 文字)

CONDITIONS データタイプ構造体のパラメータパート 2 (終了条件)

パラメータ	パラメ ータタ イプ	データタ イプ	説明
ENDCOND	IN	UInt	このパラメータはメッセージ終了条件を指定します。 • 01H - 応答時間 • 02H - メッセージ時間 • 04H - 文字間ギャップ • 08H - 最大長 • 10H - N + LEN + M • 20H - シーケンス
MAXLEN	IN	UInt	最大メッセージ長:最大長終了条件が選択された場合にのみ使用されます。 0~1023 バイト
N	IN	UInt	メッセージ内の長さフィールドのバイト位置。 N + LEN + M 終了条件でのみ使用します。 1~ 1023 バイト
LENGTHSIZE	IN	UInt	バイトフィールドのサイズ $(1, 2, \pm t)$ イト $(1, 2, \pm t)$ の $(1, 2, \pm t)$ が
LENGTHM	IN	UInt	長さフィールドの値に含まれない長さフィール ドに続く文字数を指定します。 N + LEN + M 終了条件でのみ使用します。 0~ 255 バイト
RCVTIME	IN	UInt	最初の文字を受信するために待機する時間数を 指定します。 受信操作は、指定された時間内 に文字が正しく受信されないと終了します。 応答時間条件でのみ使用されます。 0~65535 ビット時間、最大 8 秒 このパラメータは、開始条件を評価するだけな ので、終了条件として評価されるわけではあり ません。 明確な終了条件を選択する必要があ ります。

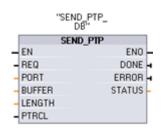
パラメータ	パラメ ータタ イプ	データタ イプ	説明
MSGTIME	IN	UInt	最初の文字が受信されてから、メッセージ全体が完全に受信されるのを待機する時間数を指定します。 このパラメータは、メッセージタイムアウト条件が選択された場合にだけ使用されます。 0~65535 ミリ秒
CHARGAP	IN	UInt	文字間のビット時間数を指定します。 文字間のビット時間数が指定された値を超えると、終了条件が満たされます。 文字間ギャップ条件でのみ使用されます。 0~65535 ミリ秒
ENDSEQ1CTL	IN	Byte	シーケンス 1 は各文字のコントロールを無視/ 比較します。 これらのコントロールは、終了シーケンスでの 各文字のイネーブルビットです。 文字 1 はビ ット 0、文字 2 はビット 1、、文字 5 はビッ ト 4 になります。文字に関連付けられたビッ トを無効にすると、このシーケンス位置で任意 の文字が一致します。
ENDSEQ1	IN	Char[5]	シーケンス 1 の開始文字(5 文字)

8.6 ポイントツーポイント命令

条件コード

STATUS (W#16#)	説明
80C0	不正な開始条件が選択されました
80C1	不正な終了条件が選択されました。終了条件が選択されていません
80C2	受信割り込みが有効にされましたが不可能です
80C3	最大長終了条件が有効にされ、最大長が 0 であるかまたは 1024 よりも 大きくなっています
80C4	計算された長さが有効にされ、N が 1023 以上になっています
80C5	計算された長さが有効にされ、長さが1、2または4ではありません
80C6	計算された長さが有効にされ、Mの値が 255 よりも大きくなっています
80C7	計算された長さが有効にされ、計算された長さが 1024 よりも大きくなっています
80C8	応答タイムアウトが有効にされ、応答タイムアウトがゼロです
80C9	文字間ギャップタイムアウトが有効にされ、0または2500よりも大きくなっています
80CA	アイドルラインタイムアウトが有効にされ、0 または 2500 よりも大き くなっています
80CB	終了シーケンスが有効にされましたが、すべての文字が「don't care」です
80CC	開始シーケンス(4 つのうちいずれか)が有効にされましたが、すべての 文字が「don't care」です

8.6.5 SEND_PTP命令



SEND_PTP (Send Point-to-Point data)がデータ送信を開始します。 SEND_PTP は指定されたバッファを CM に転送します。 CPU プログラムは、CM がデータを指定されたボーレートで送信する間、続行します。 一度に保留可能な送信操作は 1 つだけです。 CM は 2 番目の SEND_PTP が実行されるとエラーを返しますが、その間に CM はすでにメッセージを送信しています。

パラメータ	パラメー タタイプ	データタ イプ	説明
REQ	IN	Bool	この送信イネーブル入力信号の立ち上がりエッジで要求された送信を有効にします。 これにより、バッファの内容のポイントツーポイント通信モジュール(CM)への転送が開始されます。
PORT	IN	PORT	通信ポート識別子: この論理アドレスは定数で、 デフォルトタグテーブルの[Constants]タブで参 照できます。
BUFFER	IN	Variant	このパラメータは、転送バッファの開始位置を ポイントします。 ブールデータまたはブール配列はサポートされ ていません。
LENGTH	IN	UInt	送信されたフレームのバイト長 複雑な構造体を送信する場合は、常に 0 を使用 してください。
PTRCL	IN	Bool	このパラメータは、アタッチされた CM 内で実装される通常のポイントツーポイントプロトコルまたはシーメンスが提供するプロトコルとして、バッファを選択します。 FALSE = ユーザープログラム制御のポイントツーポイント操作 (唯一の有効なオプション)
DONE	OUT	Bool	1 スキャンについて、最後の要求がエラーなしで 完了した後、TRUE になります。

8.6 ポイントツーポイント命令

パラメータ	パラメー タタイプ	データタ イプ	説明
ERROR	OUT	Bool	1 スキャンについて、最後の要求がエラーで終了 した後、TRUE になります。
STATUS	OUT	Word	実行条件コード

送信操作の進行中、DONE 出力および ERROR 出力で FALSE になります。 送信操作が完了し、DONE 出力または ERROR 出力のいずれかが 1 つのスキャンサイクルで TRUE にセットされ、送信操作のステータスが表示されます。 DONE または ERROR が TRUE の場合、STATUS 出力は有効です。

この命令は、通信モジュール(CM)が送信データを受け取れば、ステータス 16#7001 を返します。 以降の SEND_PTP 実行では、CM がまだ送信中であれば 16#7002 が返されます。 送信操作が完了すると、CM はエラーが発生しなかった場合、送信操作のステータス 16#0000 を返します。 以降の REQ LOW での SEND_PTP 実行では、ステータス 16#7000(ビジーでない)が返されます。

出力値と REQ との関係:

命令は、送信プロセスのステータスを確認するため、定期的に呼び出されるものとします。 以下の図では、命令はスキャンごとに呼び出されると仮定しています(STATUS 値によって表示)。

REQ							
DONE							
DONE							
ERROR							
STATUS	7000H	7001H	7002H	7002H	7002H	0000H	7000H

以下の図は、送信操作を開始するために REQ が(1 スキャンについて)パルス化される場合、DONE パラメータと STATUS パラメータが 1 つのスキャンについてのみ有効であることを示しています。

REQ								
DONE								
ERROR								
STATUS	7000H	7001H	7002H	7002H	7002H	0000H	7000H	7000H

以下の図は、エラーが発生した場合の DONE、ERROR、STATUS パラメータの関係を示しています。

REQ								
DONE								
DONE								
ERROR								
STATUS	7000H	7001H	7002H	7002H	7002H	80D1H	7000H	7000H

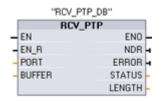
STATUS (W#16#)	説明
80D0	送信側が有効な間に新規要求がありました
80D1	待機時間内に CTS がなかったため送信が中止されました
80D2	DCE デバイスから DSR がなかったため送信が中止されました
80D3	キューオーバーフロー(1024 バイトを超える送信)のため送信が中止されました
7000	ビジーでない
7001	要求の受け入れ中でビジー(最初の呼び出し)
7002	ポーリング中でビジー(n 番目の呼び出し)

PTP_SEND に関する LENGTH パラメータと DATA パラメータの相互作用

PTP_SEND 命令で送信可能なデータの最小サイズは、1 バイトです。 DATA パラメータは、送信されるデータのサイズを指定します。 DATA パラメータに BOOL や BOOL 配列は使用できません。

LENGTH パラメ ータ	DATA パラメータ	説明
LENGTH = 0	未使用	DATA パラメータで定義された完全なデータが送信されます。 LENGTH = 0 の場合、送信されるバイト数を指定する必要はありません。
LENGTH > 0	基本データタイプ	LENGTH 値には、このデータタイプのバイト数が含まれている必要があります。 含まれていない場合、何も送信されず、エラー8088H が返されます。
	Structure	LENGTH 値は、構造体のバイト長よりも少ないバイト数を含むことができます。 この場合、最初の LENGTH バイトが送信されます。
	Array	LENGTH 値は、配列のバイト長よりも少ないバイト数を含む ことができます。 この場合、LENGTH バイト内に完全に収ま る配列要素だけが送信されます。
		LENGTH は、データ要素バイト数の倍数である必要があります。 倍数でないと、STATUS = 8088H、ERROR = 1 となり、何も送信されません。
	String	文字列フォーマットの完全なメモリレイアウトが送信されます。 LENGTH 値には、最大長、実際長、文字列文字のバイトを含める必要があります。
		STRING データタイプの場合、すべての長さと文字は 1 バイトサイズでなければなりません。
		文字列が DATA パラメータで実際のパラメータとして使用された場合、LENGTH 値にも 2 つの長さフィールドに 2 バイトを含める必要があります。

8.6.6 RCV_PTP命令



RCV_PTP (Receive Point-to-Point)は、CM に受信されたメッセージを確認します。メッセージが使用可能であれば、CM から CPU に転送されます。 エラーでは、対応する STATUS 値が返されます。

STATUS 値は、NDR または ERROR が TRUE であれば有効です。 STATUS 値は、CM での受信操作が終了した理由を示します。通常は正の値で、受信操作が成功し、受信プロセスが正常に終了したことを示します。 STATUS が負の値の場合(16 進値の最上位ビットがセットされる)、受信操作がパリティ、フレーミング、またはオーバーランエラーなどのエラー条件で終了したことを示します。

各ポイントツーポイント CM モジュールは、最大 1K バイトをバッファに格納することができます。 格納するのは 1 つの大きいメッセージまたは複数の小さいメッセージになります。

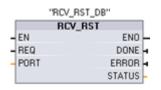
パラメー タ		データタ イプ	説明
EN_R	IN	Bool	この入力が TRUE の場合、CM モジュールに受信 メッセージがないか確認してください。 メッセー ジが正しく受信されると、モジュールから CPU に 転送されます。 EN_R が FALSE の場合、CM に受 信メッセージがないか確認され STATUS 出力がセ ットされますが、メッセージは CPU に転送されま せん。
PORT	IN	PORT	通信ポート識別子: この論理アドレスは定数で、デフォルトタグテー ブルの[Constants]タブで参照できます。
BUFFER	IN	Variant	このパラメータは、受信バッファの開始位置をポイントします。 このバッファには、最大長メッセージを受信できる十分な大きさが必要です。 ブールデータまたはブール配列はサポートされていません。
NDR	OUT	Bool	新しいデータの準備ができ、操作がエラーなしで 完了した場合、1 つのスキャンについて TRUE に なります。

8.6 ポイントツーポイント命令

パラメー タ	パラメー タタイプ	データタ イプ	説明
ERROR	OUT	Bool	1 スキャンについて、操作がエラーありで終了すると、TRUE になります。
STATUS	OUT	Word	実行条件コード
LENGTH	OUT	UInt	戻りメッセージの長さ(バイト単位)

STATUS (W#16#)	説明
0000	バッファがありません
80E0	受信バッファが不足しているためメッセージが終了しました
80E1	メッセージがパリティエラーで終了しました
80E2	メッセージがフレーミングエラーで終了しました
80E3	メッセージがオーバーランエラーで終了しました
80E4	計算された長さがバッファサイズを超えているため、メッセージが終了 しました
0094	受信した最大文字列長のためメッセージが終了しました
0095	メッセージタイムアウトのためメッセージが終了しました
0096	文字間タイムアウトのためメッセージが終了しました
0097	応答タイムアウトのためメッセージが終了しました
0098	「N+LEN+M」の長さ条件が満たされたため、メッセージが終了しました
0099	終了シーケンスが満たされたため、メッセージが終了しました

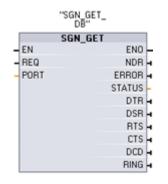
8.6.7 RCV_RST命令



RCV_RST (Receiver Reset)は CM の受信バッファをクリアします。

パラメー タ	パラメー タタイプ	データ タイプ	説明
REQ	IN	Bool	このイネーブル入力信号の立ち上がりエッジで、受信側リセットを有効にします。
PORT	IN	PORT	通信ポート識別子: このポートは、モジュールの論理アドレスを使用し て指定する必要があります。
DONE	OUT	Bool	1 つのスキャンで TRUE の場合、最後の要求はエラーなしで完了したことを示します。
ERROR	OUT	Bool	TRUE の場合、最後の要求はエラーで終了したことを示します。 また、この出力が TRUE の場合、STATUS 出力には関連するエラーコードが含められます。
STATUS	OUT	Word	エラーコード:

8.6.8 SGN_GET命令



SGN_GET (Get RS232 Signals)は、RS232 通信信号の現在の状態を読み取ります。 このファンクションは RS232 CM (通信モジュール)に関してのみ有効です。

8.6 ポイントツーポイント命令

パラメー タ	パラメー タタイプ	データ タイプ	説明
REQ	IN	Bool	この入力信号の立ち上がりエッジで RS232 信号状態 の値を取得します
PORT	IN	PORT	通信ポート識別子:
			この論理アドレスは定数で、デフォルトタグテーブルの[Constants]タブで参照できます。
NDR	OUT	Bool	新しいデータの準備ができ、操作がエラーなしで完了した場合、1 つのスキャンについて TRUE になります。
ERROR	OUT	Bool	1 スキャンについて、操作がエラーありで終了すると、TRUE になります。
STATUS	OUT	Word	実行条件コード
DTR	OUT	BOOL	データ端末レディ、モジュールレディ(出力)
DSR	OUT	BOOL	データセットレディ、通信パートナーレディ(入力)
RTS	OUT	Bool	送信要求、モジュール送信レディ(出力)
CTS	OUT	Bool	送信可、通信パートナーはデータを受信可能(入力)
DCD	OUT	Bool	データキャリア検出、受信信号レベル(常に False、 サポートされません)
RING	OUT	Bool	リングインジケータ、着信呼び出し(常に False、サポートされません)

STATUS (W#16#	説明
80F0	CM は RS485 モジュールであり、使用可能な信号がありません
80F1	ハードウェアフロー制御のため信号を設定できません
80F2	モジュールが DTE のため DSR を設定できません
80F3	モジュールが DCE のため DTR を設定できません

8.6.9 SGN_SET命令



SGN_SET (Set RS232 Signals)は RS232 通信信号の状態を 設定します。 このファンクションは RS232 CM (通信モジュ ール)に関してのみ有効です。

パラメー タ		データタ イプ	説明
REQ	IN	Bool	この入力信号の立ち上がりエッジで、設定された RS232 信号操作を開始します
PORT	IN	PORT	通信ポート識別子:
			この論理アドレスは定数で、デフォルトタグテーブルの[Constants]タブで参照できます。
SIGNAL	IN	Byte	設定する信号を選択します (複数選択可)。
			• 01H = RTS を設定
			• 02H = DTR を設定
			• 04H = DSR を設定
RTS	IN	Bool	送信要求、モジュール設定する値の送信レディ (True または False)
DTR	IN	Bool	データ端末レディ、モジュール設定する値の送信 レディ(True または False)
DSR	IN	Bool	データセットレディ(DCE タイプインターフェース にのみ適用) (使用しません)
DONE	OUT	Bool	1 スキャンについて、最後の要求がエラーなしで 完了した後、TRUE になります。
ERROR	OUT	Bool	1スキャンについて、最後の要求がエラーで終了 した後、TRUEになります。
STATUS	OUT	Word	実行条件コード

STATUS (W#16#)	説明
80F0	CM は RS485 モジュールで設定可能な信号がありません
80F1	ハードウェアフロー制御のため信号を設定できません
80F2	モジュールが DTE のため DSR を設定できません
80F3	モジュールが DCE のため DTR を設定できません

8.7 エラー

ポイントツーポイント命令の戻り値

各ポイントツーポイント命令には、完了ステータスを示す3つの出力があります。

パラメータ	データタイ プ	デフォルト	説明
DONE	Boolean	FALSE	1 つのスキャンで TRUE の場合、最後の要求 はエラーなしで完了したことを示します。
ERROR	Boolean	FALSE	TRUE は、最後の要求がエラー(STATUS で適用可能なエラーコード)で終了したことを示します。
STATUS	Word	0	適用可能な場合、エラークラスとエラー番号を含む2バイト。 STATUS には、ファンクション実行の時間値が保持されます。

一般的なエラークラスとエラー

クラスの説明	エラークラス	説明
ポートコンフィグレーション	80Ax	一般的なポートコンフィグレーションエラ ーの定義に使用します
送信コンフィグレーション	80Bx	一般的な送信コンフィグレーションエラー の定義に使用します
受信コンフィグレーショ ン	80Cx	一般的な受信コンフィグレーションエラー の定義に使用します
送信ランタイム	80Dx	一般的な送信ランタイムエラーの定義に使 用します
受信ランタイム	80Ex	一般的な受信ランタイムエラーの定義に使 用します
信号処理	80Fx	すべての信号処理に関連する一般的なエラ ーの定義に使用します

ポートコンフィグレーションエラー

イベント/エラー ID	説明
0x80A0	特定のプロトコルが存在しません
0x80A1	特定のボーレートが存在しません
0x80A2	特定のパリティが存在しません
0x80A3	特定のデータビット数が存在しません
0x80A4	特定のストップビット数が存在しません
0x80A5	特定のフロー制御タイプが存在しません

送信コンフィグレーションエラー

イベント/エラー ID	説明
0x80B0	特定のプロトコルが存在しません
0x80B1	特定のボーレートが存在しません
0x80B2	特定のパリティが存在しません
0x80B3	特定のデータビット数が存在しません
0x80B4	特定のストップビット数が存在しません
0x80B5	特定のフロー制御タイプが存在しません

受信コンフィグレーションエラー

イベント/エラー ID	説明
0x80C0	開始条件エラー
0x80C1	終了条件エラー
0x80C3	最大長エラー
0x80C4	N 値エラー(N+LEN+M を参照)
0x80C5	長さサイズエラー(MAXLEN または N+LEN+M を参照)
0x80C6	M 値エラー(N+LEN+M を参照)
0x80C7	N の長さ M 値エラー(N+LEN+M を参照)
0x80C8	応答タイムアウトエラー、指定された受信時間中にメッセージが 受信されませんでした (RCVTIME または MSGTIME を参照)。
0x80C9	文字間タイムアウトエラー(CHARGAP を参照)
0x80CA	アイドルラインタイムアウトエラー(アイドルラインを参照)
0x80CB	指定された終了シーケンスが、すべて「don't care」文字で設定されました
0x80CC	指定された開始シーケンスが、すべて「don't care」文字で設定されました

信号エラー

イベント/エラー ID	説明
0x80F0	通信モジュールは RS485 モジュールであり、使用可能な信号がありません
0x80F1	通信モジュールは RS232 モジュールですが、ハードウェアフロー 制御が有効になっているため信号を設定できません
0x80F2	モジュールが DTE デバイスのため、DSR 信号は設定できません

送信ランタイムエラー

イベント/エラー ID	説明
バッファ制限	CP の使用可能な送信バッファ合計を超過しました
0x80D0	送信側が有効な間に新しい要求を受信しました
0x80D1	受信側が、有効な送信を保留するためにフロー制御要求を発行し、指定された待機時間中に送信を再び有効にしませんでした。
	このエラーは、ハードウェアフロー制御中に、受信側が指定された待機時間内に CTS をアサートしなかった場合にも生成されます
0x80D2	送信要求は、DCE から DSR 信号が受信されなかったため中止されました
0x80D3	CP の使用可能な送信バッファ合計を超過しました
0x7000	送信ファンクションがビジーではありません
0x7001	送信ファンクションが最初の呼び出しでビジーです
0x7002	送信ファンクションが以降の呼び出しでビジーです(最初の呼び出 し後のポーリング)

受信ランタイム戻り値

イベント/エラー ID	説明
0x80E0	受信バッファが不足しているためメッセージが終了しました
0x80E1	パリティエラーのためメッセージが終了しました
0x80E2	フレーミングエラーのためメッセージが終了しました
0x80E3	オーバーランエラーのためメッセージが終了しました
0x80E4	指定された長さがバッファサイズ合計を超過しているため、メッ セージが終了しました
0x0094	最大文字長(MAXLEN)を受信したため、メッセージが終了しました
0x0095	指定時間(MSGTIME)内に完全なメッセージが受信されなかったため、メッセージが終了しました
0x0096	文字間時間(CHARGAP)中に次の文字が受信されなかったため、メッセージが終了しました
0x0097	指定時間(RCVTIME)内に最初の文字が受信されなかったため、メッセージが終了しました
0x0098	「N+LEN+M」の長さ条件が満たされたため、メッセージが終了しました(N+LEN+M)
0x0099	終了シーケンス(ENDSEQ)が満たされたため、メッセージが終了 しました

その他のパラメータエラー

イベント/エラー ID	説明
0x8n3A	パラメータnで不正なポインタが用意されました
0x8070	すべての内部インスタンスメモリが使用中です
0x8080	ポート番号が無効です
0x8082	パラメータ設定は、パラメータ設定がバックグラウンドですでに 進行中のため失敗しました
0x8083	バッファオーバーフローです。 CM が許可されるよりも多くのデータを返しました。
0x8085	LEN パラメータの値が 0 であるか、または最大許容値よりも大きくなっています。
0x8088	LEN パラメータが、DATA で指定されているメモリ領域よりも大きくなっています。

オンライン診断ツール

9.1 ステータス **LED**

CPU と I/O モジュールは LED を使用して、モジュールまたは I/O の動作ステータスに 関する情報を提供します。CPU には以下のステータスインジケータがあります。

STOP/RUN

- 橙の点灯は STOP モードを示します
- 緑の点灯は RUN モードを示します
- 点滅(緑と橙の変化)は CPU が起動中であることを示します

ERROR

- 赤の点滅は、CPU 内部エラー、メモリカードエラー、またはコンフィグレーションエラー(不一致のモジュール)などのエラーを示します。
- 赤の点灯はハードウェア故障を示します
- MAINT (メンテナンス)メモリカードを挿入すると常に点滅します。 すると CPU は STOP モードに切り替わります。 CPU が STOP モードに切り替わった後、以下の いずれかのファンクションを実行して、メモリカードの評価を開始してください。
 - CPUをRUNモードに切り替える
 - メモリリセットを実行する(MRES)
 - CPU の電源を切った後に再投入する

説明	STOP/RUN 橙 / 緑	ERROR 赤	MAINT 橙
電源はオフです	オフ	オフ	オフ
起動、セルフテ スト、ファーム	点滅 (橙と緑で切り替え)	-	オフ
ウェアの更新			
STOP モード	オン (橙)	-	-
RUN モード	オン (橙) 緑	-	-

9.1 ステータス LED

説明	STOP/RUN 橙 / 緑	ERROR 赤	MAINT 橙
メモリカードを	オン	-	点滅
取り外します	(橙)		
エラー	オン (橙または緑)	点滅	-
メンテナンスが 要求されました	オン (橙または緑)	-	オン
ハードウェア故 障	オン (橙)	オン	オフ
LED テストま たは CPU ファ ームウェア故障	点滅 (橙と緑で切り替え)	点滅	点滅

CPU にも **PROFINET** 通信のステータスを示す **2** つの **LED** があります。 底部のフロントカバーを開けて **PROFINET LED** を確認してください。

- Link (緑)の点灯は正しく接続されたことを示します
- Rx/Tx (黄)の点灯は送信活動を示します

CPU と各デジタルシグナルモジュール(SM)には、各デジタル入出力の I/O Channel LED があります。 I/O Channel (緑)のオンオフは、個々の入出力状態を示します。

さらに、各デジタル SM には、モジュールのステータスを示す DIAG LED があります。

- 緑はモジュールが運転可能であることを示します
- 赤はモジュールが故障しているか運転不可能であることを示します

各アナログ SM には、各アナログ入出力の I/O Channel LED があります。

- 緑はチャンネルが設定され有効であることを示します
- 赤は個々のアナログ入力または出力のエラー条件を示します

さらに、各アナログ SM には、モジュールのステータスを示す DIAG LED があります。

- 緑はモジュールが運転可能であることを示します
- 赤はモジュールが故障しているか運転不可能であることを示します

SM はモジュールの電力のありなしを検出します(必要に応じてフィールド側電力)。

説明	DIAG (赤 / 緑)	I/O Channel (赤 / 緑)
フィールド側電源はオフです	赤の点滅	赤の点滅
未設定または更新中	緑の点滅	オフ
モジュールはエラーなしで設定済み	オン(緑)	オン(緑)
エラー条件	赤の点滅	-
I/O エラー(診断が有効な状態)	-	赤の点滅
I/O エラー(診断が無効な状態)	-	オン(緑)

9.2 オンラインで CPU に接続

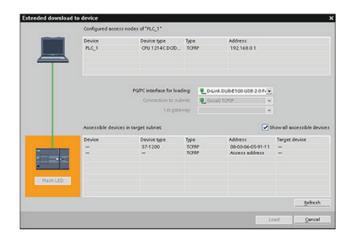
プログラミングデバイスとターゲットシステム間のオンライン接続は、プログラムと プロジェクトエンジニアリングデータのターゲットシステムへのロード、また以下の ような作業にも必要です。

- ユーザープログラムのテスト
- CPU の動作モードの表示および変更
- CPUの日付と時刻の表示および設定
- モジュール情報の表示
- オンラインブロックとオフラインブロックの比較
- ハードウェアの診断

[Online tools]タスクカードを使用して、オンラインまたは診断表示でターゲットシステムのデータにアクセスできます。



9.3 IP アドレスと時刻の設定



デバイスの現在のオンライン ステータスが、プロジェクト ナビゲーションでデバイスの 右隣にアイコンによって表示 されます。

橙色はオンライン接続を示し ます。

[Accessible Nodes]を選択して、ネットワーク上の CPU を検索します。

💋 Go online

[Go online]をクリックして、ネットワーク上の CPU に接続します。

9.3 IP アドレスと時刻の設定

オンライン CPU の IP アドレスと時刻を 設定することができます。

[Online & diagnostics]エリアからオンライン CPU に接続した後、IP アドレスを表示または変更することができます。

詳細については、IPアドレス (ページ 99) に関するセクションを参照してください。

オンライン CPU の日付と時刻のパラメータを表示または設定することもできます。





9.4 オンライン CPU の CPU オペレータパネル

[CPU operator panel]タスクカードには、オンライン CPU の動作モード(STOP または RUN)が表示されます。 このパネルには、CPU にエラーが発生しているか、または値が強制されているかどうかも表示されます。 CPU 操作パネルを使用して、オンライン CPU の動作モードを変更します。

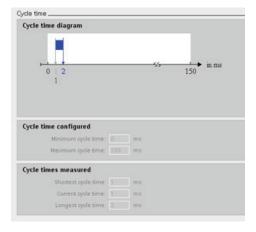
▼ CPU operator p	anel
PLC_1 [CPU 1214C1	palpalpc]
RUN / STOP	RUN
☐ ERROR	STOP
MAINT	MRES

9.5 サイクルタイムとメモリ使用率のモニタ

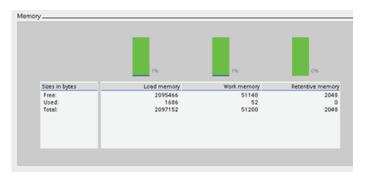
オンライン CPU のサイ クルタイムとメモリ使用 率をモニタすることがで きます。

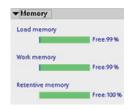
オンライン CPU に接続 した後、以下の測定値を 表示できます。

- サイクルタイム
- メモリ使用率







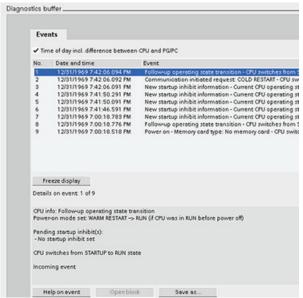


9.6 CPU での診断イベントの表示

9.6 CPU での診断イベントの表示

診断バッファを使用して、CPU の最近の活動を確認することができます。 診断バッファには次のエントリがあります。

- 診断イベント
- CPU 動作モード(STOP または RUN モードへの変更)



先頭のエントリは直前に発生したイベントです。 診断バッファの各エントリには、イベントが記録された日付と時刻、および説明があります。 エントリの最大数は CPU に依存します。 最大 50 のエントリがサポートされています。

診断バッファ内の最新の 10 イベントだけが、永続的に保存されます。 CPU を工場出荷時の設定にリセットすると、エントリが削除され診断バッファがリセットされます。

9.7 ユーザープログラムのモニタテーブル

モニタテーブルを使って、CPU がユーザープログラムを実行するとき、データポイントでファンクションをモニタおよび制御することができます。 これらのデータポイントは、モニタファンクションまたは制御ファンクションに応じて、プロセスイメージ(Iまたは Q)、物理(I:Pまたは Q:P)、M、あるいは DB になります。

モニタファンクションは、プログラムシーケンスを変更しません。 CPU 内のプログラムシーケンス情報とプログラムのデータを表示します。

制御ファンクションによっては、プログラムのシーケンスとデータを制御することができます。 制御ファンクションを使用する場合は注意が必要です。 これらのファンクションは、ユーザープログラム/システムプログラムの実行に重大な影響を及ぼす可能性があります。 3 つの制御ファンクションは、修正、強制、STOP での出力の有効化です。モニタテーブルを使って、以下のオンラインファンクションを実行できます。

- タグステータスのモニタ
- 個々のタグの値修正
- タグを特定値に強制

タグのモニタと修正のタイミングを選択します。

- スキャンサイクルの先頭 スキャンサイクルの先頭で値の読み書きをします
- スキャンサイクルの最後 スキャンサイクルの最後で値の読み書きをします
- 停止に切り替え

モニタテーブルを作成するには

- 1. [Add new watch table] をダブルクリックして、新 しいモニタテーブルを開きます。
- 2. タグ名を入力し、モニタテーブルにタグを追加します。

タグのモニタでは以下のオプションが使用できます。

- すべてモニタ: 有効なモニタテーブルで、表示されているタグのモニタを開始します。
- モニタ有効化: 有効なモニタテーブルで、表示されているタグのモニタを開始します。 モニタテーブルは、直ちにタグをモニタします。



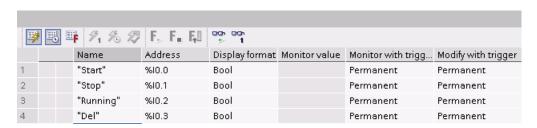
9.7 ユーザープログラムのモニタテーブル

タグの修正では以下のオプションが使用できます。

- [Modify to 0]では選択されたアドレスの値を「0」に設定します。
- [Modify to 1]では選択されたアドレスの値を「1」に設定します。
- [Modify now]では、1 つのスキャンサイクルで選択されたアドレスの値をただちに変 更します。
- [Modify with trigger]では、選択されたアドレスの値を変更します。 このファンクションでは、選択されたアドレスが実際に修正されたかどうかを示す フィードバックはありません。 変更のフィードバックが必要な場合は、[Modify now]ファンクションを使用します。
- [Enable peripheral outputs]は、コマンド出力無効化を無効にし、CPU が STOP モードのときだけ使用できます。

タグをモニタするには、CPU へのオンライン接続が必要です。

ィールドのデータポイントの実際値が表示されます。



モニタテーブルの上部にあるボタンを使って、各種ファンクションを選択できます。 モニタ対象のタグ名を入力し、ドロップダウンから表示フォーマットを選択します。 CPU にオンライン接続した状態で[Monitor]ボタンをクリックすると、[Monitor value]フ

PLC タグのモニタまたは修正時のトリガ使用

トリガは、スキャンサイクルで選択されたアドレスがどの時点でモニタまたは修正されるかを指定します。

トリガタイプ	説明
定常的	継続的にデータを収集
スキャンサイク ルの開始時	定常的: CPU が入力を読み込んだ後、スキャンサイクルの開始時 にデータを継続的に収集します
	1回 CPU が入力を読み込んだ後、スキャンサイクルの開始時にデータを 1回収集します
スキャンサイク ルの最後	定常的: CPU が出力を書き出した後、スキャンサイクルの終了時 にデータを継続的に収集します
	1回 CPU が出力を書き出した後、スキャンサイクルの終了時にデータを 1回収集します
STOP への移行	定常的: CPU の STOP への移行時にデータを継続的に収集します
時	1回 CPUの STOP への移行後にデータを 1回収集します

任意のトリガで PLC タグを修正する場合は、サイクルの開始または終了を選択します。

• 出力の修正: 出力の修正に最適なトリガイベントは、CPU が出力を書き出す直前の、スキャンサイクルの終了時です。

スキャンサイクルの開始時に出力値をモニタして、物理出力に書き込まれる値を判別します。また、CPU が物理出力に値を書き込む前に出力をモニタして、プログラムロジックを確認し、実際の入出力動作を比較します。

• 入力の修正: 入力の修正に最適なトリガイベントは、CPU が入力を読み取る直後、かつプログラムが入力値を使用する前の、スキャンサイクルの開始時です。

スキャンサイクルの開始で入力を修正する場合は、スキャンサイクルの終了時の入力値もモニタして、入力値がスキャンサイクルの開始時から変更されていないことを確認する必要があります。 値が開始時と終了時で異なる場合は、間違ってユーザープログラムの入力へ書き込みを行っている可能性があります。

CPU が STOP に移行した可能性の理由を診断するには、[Transition to STOP]トリガを使用して、最後の処理値を取得してください。

9.7 ユーザープログラムのモニタテーブル

STOP モードでの出力の有効化

モニタテーブルを使用して、CPU が STOP モードのときに出力に書き込むことができます。 この機能で出力の配線を確認し、接続されたプロセスデバイスの端子に対して、出力ピンに接続された配線が HIGH または LOW の信号を開始することを検証できます。

个警告

CPU が STOP モードであっても、物理出力を有効にすれば、接続されているプロセスポイントを有効にすることができます。

出力が有効になっているとき、STOP モードで出力の状態を変更できます。 出力が無効になっていれば、STOP モードで出力を修正することはできません。

- STOP モードで出力の修正を可能にするには、[Online]メニューの[Modify]コマンド の[Enable peripheral outputs]を選択するか、モニタテーブルの対応する行を右クリックします。
- CPU を RUN モードに設定すると、[Enable peripheral outputs]オプションは無効になります。
- 強制されている入力または出力がある場合、CPU は STOP モードで出力を有効に することができません。 最初に強制ファンクションをキャンセルする必要があります。

CPU での強制値

CPUでは、物理入力アドレスまたは出力アドレス(I_:P または Q_:P)をモニタテーブルで指定して、入出力ポイントを強制することで、強制を開始することができます。

プログラムでは、強制値で上書きされた物理入力を読み取ります。 プログラムはこの 強制値を使って処理を行います。 プログラムが物理出力を書き込むと、出力値は強制 値によって上書きされます。 強制値は物理出力として表示され、処理に使用されます。

モニタテーブルで入力または出力が強制されている場合、強制操作はユーザープログラムの一部になります。 プログラミングソフトウェアを閉じた後も、選択された強制ファンクションは動作中の CPU プログラムで有効のままになります。無効になるのは、プログラミングソフトウェアでオンライン接続して消去され、強制ファンクションが停止した場合です。 強制ポイントを持ったプログラムをメモリカードから別の CPU にロードすると、プログラムで選択されたポイントが引き続き強制されます。

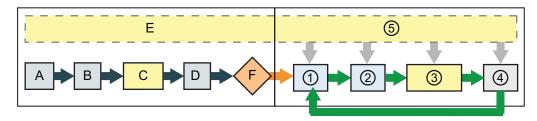
CPU が書き込み保護されたメモリカードからユーザープログラムを実行している場合、モニタテーブルから入出力の強制を開始または変更することはできません。書き込み保護されたユーザープログラムの値を上書きすることができないためです。書き込み保護された値を強制しようとすると、エラーが発生します。ユーザープログラムの転送にメモリカードを使用する場合、そのメモリカード上にあるすべての強制要素はCPUに転送されます。

注記

HSC、PWM、PTO に割り付けられたデジタル I/O ポイントは強制できません

高速カウンタ(HSC)、パルス幅変調(PWM)、パルストレイン出力(PTO)デバイスで使用されるデジタル I/O ポイントは、デバイスコンフィグレーション中に割り付けられます。 これらのデバイスにデジタル I/O ポイントのアドレスが割り付けられると、割り付けられた I/O ポイントのアドレスをモニタテーブルの強制ファンクションで変更することはできません。

9.7 ユーザープログラムのモニタテーブル



起動

- A Iメモリ領域をクリアしても、強制ファンクションの影響は受けません。
- B 出力値の初期化は、強制ファンクションの影響を受けません。
- C スタートアップ OB の実行中に、CPU は、ユーザープログラムが物理入力に アクセスすると、強制値を適用しま す。
- D キューへの割り込みイベントの保存 は、影響を受けません。
- E 出力への書き込み有効化は、影響を受けません。

RUN

- ① Qメモリを物理出力に書き込む間、出力が更 新されると CPU は強制値を適用します。
- ② 物理入力を読み取るとき、CPUは、入力をI メモリにコピーする直前に強制値を適用しま す。
- ③ ユーザープログラムの実行中に(プログラムサイクル OB)、CPU は、ユーザープログラムが物理入力にアクセスするか、または物理出力を書き込むと、強制値を適用します。
- ④ 通信要求とセルフテスト診断の処理は、強制 ファンクションの影響を受けません。
- ⑤ スキャンサイクルのどの部分での割り込み処理であっても、影響は受けません。

技術仕様

A.1 一般技術仕様

規格への準拠

S7-1200 オートメーションシステムは、以下の規格およびテスト仕様に準拠しています。 **S7-1200** オートメーションシステムのテスト基準は、これらの規格およびテスト仕様に 基づいています。

CE 承認



S7-1200 オートメーションシステムは、以下に記載する **EC** 指令に基づく要件および安全に関連する目的を満たしており、欧州共同体官報に記載されているプログラマブルコントローラの欧州整合規格(**EN**)に適合しています。

- EC 指令 2006/95/EC(低電圧指令)『特定の電圧限度内で使用するように設計された 電気機器』
 - EN 61131-2:2007 プログラマブルコントローラ 機器要件およびテスト
- EC 指令 2004/108/EC (EMC 指令)『電磁両立性』
 - 放出規格

EN 61000-6-4:2007: 産業環境

- 電磁波耐性規格

EN 61000-6-2:2005: 産業環境

- EC 指令 94/9/EC (ATEX)『爆発性雰囲気での使用を目的とした機器および保護システム』
 - EN 60079-15:2005: 保護のタイプ'n'

CE 適合宣言は、以下の住所でファイルに収録し所轄官庁に開示可能です。

Siemens AG

IA AS RD ST PLC Amberg Werner-von-Siemens-Str. 50 D92224 Amberg Germany

A.1 一般技術仕様

CULUS 認可



Underwriters Laboratories Inc.以下に適合

- Underwriters Laboratories Inc. UL 508 記載(工業用制御装置)
- Canadian Standards Association: CSA C22.2 Number 142 (プロセス制御機器)

通知

SIMATIC S7-1200 シリーズは CSA 規格に適合しています。

cULus ロゴは、S7-1200 が Underwriters Laboratories (UL)によって、UL 508 規格および CSA 22.2 No. 142 規格に適合していることが、試験および認証されたことを示しています。

FM 承認



Factory Mutual Research (FM):

承認規格クラス番号 3600 および 3611

以下での使用が承認済み:

クラス I、ディビジョン 2、ガスグループ A、B、C、D、温度クラス

T4A Ta = 40° C

クラス I、ゾーン 2、IIC、温度クラス T4 Ta = 40° C

ATEX 承認



EN 60079-0:2006: 爆発性雰囲気 - 一般要件

EN 60079-15:2005: 爆発性雰囲気での使用を目的とした電気機器 保護タイプ'n'

II 3 G Ex nA II T4

S7-1200 の安全な使用のため以下の特殊条件に従う必要があります。

- モジュールは、EN 60529 に従い IP54 の最小限度の保護が提供される適切な囲いの中に設置し、機器が使用される環境条件を考慮すること。
- 定格条件で温度がケーブル差し込み口で **70°C** を超えるか、または導体の分岐点で **80°C** を超える場合、選択したケーブルの温度仕様は、実際に測定した温度と適合していなければなりません。
- 定格電圧が、40%を超える過渡妨害によって超過しないように準備する必要がある。

C-Tick 承認



S7-1200 オートメーションシステムは、**AS/NZS 2064 (**クラス **A)**の規格要件 を満たしています。

海事承認

S7-1200 製品は、特定の市場や用途に関する専門機関の承認を得るため定期的に提出されています。 部品番号による正確な承認の最新リストに関して詳細情報が必要な場合は、貴社のシーメンス担当者にご相談ください。

船級協会:

- ABS (American Bureau of Shipping)
- BV (Bureau Veritas)
- DNV (Det Norske Veritas)
- GL (Germanischer Lloyd)
- LRS (Lloyds Register of Shipping)
- Class NK (日本海事協会)

A.1 一般技術仕様

産業環境

S7-1200 オートメーションシステムは、産業環境での使用向けに設計されています。

用途	騒音放射要件	耐ノイズ性要件
産業	EN 61000-6-4:2007	EN 61000-6-2:2005

電磁環境両立性(EMC)

電磁環境両立性(EMC)は、電磁環境で意図したとおりに動作し、付近にある他の電気装置を妨害する可能性がある電磁妨害(EMI)の放射レベルなしで動作する、電気装置の能力です。

電磁環境両立性 - EN 61000-6-2 による電磁波耐性		
EN 61000-4-2 静電放電	全表面への 8 kV 空中放電 露出導電面への 6 kV 接触放電	
EN 61000-4-3 放射電磁界	80~1000 MHz、10 V/m、1 kHz で 80% AM 1-4~2.0 GHz、3 V/m、1 kHz で 80% AM 2.0~2.7 GHz、1 V/m、1 kHz で 80% AM	
EN 61000-4-4 ファストトランジェントバースト	2 kV、5 kHz、AC および DC システム電源への結合ネットワーク付き 2 kV、5 kHz、I/O への結合クランプ付き	
EN 6100-4-5 サージ耐性	AC システム - 2 kV コモンモード、1kV 差動モード DC システム - 2 kV コモンモード、1kV 差動モード DC システム(I/O 信号、DC 電源システム)には外部保護が必要です。	
EN 61000-4-6 伝導妨害	150 kHz~80 MHz、10 V RMS、1kHz で 80% AM	
EN 61000-4-11 電圧低下	AC システム 1 サイクルで 0%、12 サイクルで 40%、30 サイクルで 70%(60 Hz)	

電磁環境両立性 - EN 61000-6-4 による伝導性放射および放射妨害波		
伝導性放射		
EN 55011、クラス A、グループ 1		
0.15 MHz∼0.5 MHz	<79dB (μV)準尖頭値、<66 dB (μV)平均	
0.5 MHz∼5 MHz	<73dB (μV)準尖頭値、<60 dB (μV)平均	
5 MHz∼30 MHz	<73dB (μV)準尖頭値、<60 dB (μV)平均	
放射妨害波		
EN 55011、クラス A、グループ 1		
30 MHz∼230 MHz	<40dB (μV/m)準尖頭値、10m で測定	
230 MHz∼1 GHz	<47dB (μV/m)準尖頭値、10m で測定	

環境条件

環境条件 - 輸送および保管	
EN 60068-2-2、テスト Bb、乾燥熱および EN 60068-2-1、テスト Ab、低温	-40° C∼+70° C
EN 60068-2-30、テスト Db、湿気熱	25° C~55° C、95%湿度
EN 60068-2-14、テスト Na、温度衝撃	-40° C to +70° C、滞留時間 3 時間、2 サイクル
EN 60068-2-32、自由落下	0.3 m、5 回、製品梱包
大気圧	1080~660h Pa (高度-1000~3500m に相当)

環境条件 - 動作	
周囲温度範囲 (装置下 25 mm 流入空気)	0° C~55° C 水平設置 0° C~45° C 垂直設置 95%結露なし湿度
大気圧	1080~795h Pa (高度-1000~2000m に相当)
不純物濃度	S0 ₂ : < 0.5 ppm 未満、H ₂ S: < 0.1 ppm;RH < 60%、結露な し
EN 60068-2-14、テスト Nb、温度変化	5° C~55° C、3° C/分

A.1 一般技術仕様

環境条件 - 動作	
EN 60068-2-27 機械的衝撃	15 G、11 ms パルス、各 3 軸で 6 衝撃
EN 60068-2-6 正弦波振動	DIN レールマウント: 5~9 Hz から 3.5 mm、9~150 Hz から 1G パネルマウント: 5~9 Hz から 7.0 mm、9~150 Hz から 2G 各軸 10 スイープ、1 オクターブ/分

高電位隔離テスト	
24 V / 5 V 公称回路	DC 520 V (光分離境界のタイプテスト)
115 / 230 V 接地回路	AC 1,500 V ルーチンテスト / DC 1950 V タイプテスト
115 / 230 V 回路~115 / 230 V 回路	AC 1,500 V ルーチンテスト / DC 1950 V タイプテスト
115 V / 230 V 回路~24 V / 5 V 回路	AC 1,500 V ルーチンテスト / DC 3250 V タイプテスト

保護クラス

● 保護クラス II、EN 61131-2 による(保護導体不要)

保護等級

- IP20 機械的保護、EN 60529
- 標準プローブでテストした、高電圧への指接触に対する保護。 直径 12.5mm 未満のほこり、汚れ、水、異物に対する外部保護が必要です。

定格電圧

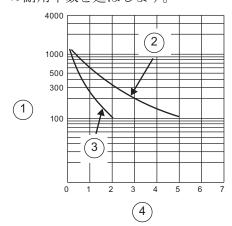
定格電圧	許容差
DC 24 V	DC 20.4 V~28.8 V
AC 120/230 V	AC 85 V∼AC 264 V、47∼63 Hz

通知

機械的接触により S7-1200 CPU、または任意のデジタルシグナルモジュールの出力電源がオンになると、約50マイクロ秒間、「1」信号がデジタル出力に送信されます。短時間パルスに応答する装置を使用している場合は特に、この件に備える必要があります。

リレーの電気耐用年数

リレーメーカー提供の一般的な性能データを以下に示します。 実際の性能は、個々の 用途に応じて異なる可能性があります。 電気負荷に適合された外部保護回路は、接点 の耐用年数を延ばします。



- ① 耐用年数(x 10³動作)
- ② AC 250 V 負荷抵抗、 DC 30 V 負荷抵抗
- ③ AC 250 V 誘導負荷(p.f=0.4) DC 30 V 誘導負荷(L/R=7ms)
- ④ 定格動作電流(A)

A.2 CPU

A.2.1 CPU 1211C仕様

技術仕様			
モデル	CPU 1211C AC/DC/リレー	CPU 1211C DC/DC/ リレー	CPU 1211C DC/DC/DC
注文番号(MLFB)	6ES7 211-1BD30- 0XB0	6ES7 211-1HD30- 0XB0	6ES7 211-1AD30- 0XB0
全般			
外形寸法 W x H x D (mm)	90 x 100 x 75		
重量	420 グラム	380 グラム	370 グラム
電力損失	10 W	8 W	
使用可能電流(CM バス)	最大 750 mA (DC 5 V)	
使用可能電流(DC 24 V)	最大 300 mA(センサ電	電源)	
デジタル入力消費電流(DC 24 V)	4 mA/入力		
CPU の機能			
ユーザーメモリ	25 キロバイト作業メモリ / 1 メガバイトロードメモリ / 2 キロバイト保持メモリ		
オンボードデジタル I/O	6入力/4出力		
オンボードアナログ 1/0	2入力		
プロセスイメージサイズ	入力 1024 バイト(I) /出力 1024 バイト(Q)		
ビットメモリ(M)	4096 バイト		
拡張モジュール数	なし		
拡張ボード数	最大 1 SB		
通信モジュール拡張	最大 3 CM		
高速カウンタ	合計 3 単相: 100 kHz (3 チャンネル) 差動: 80 kHz (3 チャンネル)		
パルス出力	2		
パルスキャッチ入力	6		
時間遅延 / 周期割り込み	1 ms 解像度で合計 4		

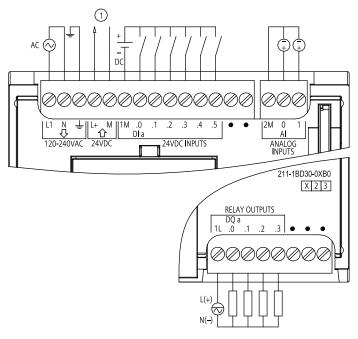
技術仕様			
モデル	CPU 1211C AC/DC/リレー	CPU 1211C DC/DC/ リレー	CPU 1211C DC/DC/DC
エッジ割り込み	6 立ち上がりおよび 6 ドで 10)	立ち下がり(10 およびオ	プションのシグナルボー
メモリカード	SIMATIC メモリカー	ド(オプション)	
リアルタイムクロックの精度	+/- 60 秒/月		
リアルタイムクロック保持時間	40°C で通常 10 日/最/	小6日(メンテナンスフリ	リーの大容量キャパシタ)
パフォーマンス			
ビット演算実行速度	0.1 μs / 命令		
Move Word 実行速度	12 μs / 命令		
実数演算実行速度	18 μs / 命令		
通信			
ポートの数	1		
タイプ	イーサネット		
接続	 HMI×3 プログラミング装置×1 ユーザープログラムのイーサネット命令×8 CPU間×3 		
 データ速度	10/100 Mb/秒		
絶縁(PLC ロジックへの外部信 号)	絶縁済み変圧器、DC	1500 V	
ケーブルタイプ	CAT5e シールドあり		
電源			
電圧範囲	AC 85∼264 V	DC 20.4~28.8 V	
回線周波数	47 Hz∼63 Hz		
入力電流			
CPU が最大負荷の場合のみ	AC 120 V / 60 mA AC 240 V / 30 mA	DC 24 V / 300 mA	
CPU がすべての拡張付属品を	AC 120 V / 180 mA	DC 24 V / 900 mA	
装備し、最大負荷の場合	AC 240 V / 90 mA		
突入電流(最大)	AC 264 V / 20 A	DC 28.8 V / 12 A	
絶縁(入力電源からロジック)	AC 1500 V	絶縁なし	
漏電、AC ラインから機能接地	最大 0.5 mA	-	

技術仕様			
モデル	CPU 1211C AC/DC/リレー	CPU 1211C DC/DC/ リレー	CPU 1211C DC/DC/DC
持続時間(停電時)	AC 120 V / 20 ms	DC 24 V / 10 ms	
	AC 240 V / 80 ms		
内部ヒューズ、ユーザー交換不	3 A、250 V、スローブ	`ロー	
可			
センサ電源	T	1	
電圧範囲	DC 20.4~28.8 V	L± DC 4 V (最小)	
出力電流定格(最大)	300 mA (短絡保護)		
最大リップルノイズ(10 MHz 未 満)	1 V 未満ピークツーピ	入力ラインと同じ	
絶縁(CPU ロジックからセンサ 電源)	絶縁なし		
デジタル入力			
入力数	6		
タイプ	シンク/ソース(IEC タィ	イプ1シンク)	
定格電圧	DC 24 V / 4 mA、(公科	۲)	
常時許容電圧	DC 30 V (最大)		
サージ電圧	DC 35 V / 0.5 秒		
ロジック 1 信号(最小)	DC 15 V / 2.5 mA		
ロジック 0 信号(最大)	DC 5 V / 1 mA		
絶縁(フィールド側からロジック)	AC 500 V / 1 分間		
・	1		
フィルタ時間	0.2、0.4、0.8、1.6、3	3.2、6.4、12.8 ms (4 グ	 ループで選択可能)
HSC クロック入力速度(最大)	単相: 100 kHz	·	,
(ロジック 1 レベル = DC 15~	差動: 80 kHz		
26 V)			
同時入力数	6		
ケーブル長(メートル)	500m シールドあり、3 ルドあり	300m シールドなし、50)m は HSC 入力用にシー
アナログ入力			
入力数	2		
タイプ	電圧(シングルエンド)		

技術仕様				
モデル	CPU 1211C AC/DC/リレー	CPU 1211C DC/DC/ リレー	CPU 1211C DC/DC/DC	
範囲	0∼10 V			
フルスケールレンジ(データワー ド)	0~27648 (「電圧の	アナログ入力表現」を参照	照)	
オーバーシュート範囲 (データワ ード)	27,649~32,511 (「i	電圧のアナログ入力表現」	を参照)	
オーバーフロー(データワード)	32,512~32767(「電	『圧のアナログ入力表現』	を参照)	
解像度	10 ビット			
最大耐電圧	DC 35 V			
平滑化	なし、弱、中、また 応答時間」を参照)	は強(ステップ応答時間に	ついては「アナログ入力	
ノイズ除去	10、50、または60日答時間」を参照)	Hz (サンプルレートについ	いては「アナログ入力応	
インピーダンス	100 ΚΩ 以上			
絶縁(フィールド側からロジッ ク)	なし			
精度(25°C / 0∼55°C)	フルスケールの 3.0%	% /3.5%		
コモンモード除去	40 dB、DC∼60 Hz			
動作信号範囲	信号およびコモンモ なければなりません	信号およびコモンモード電圧は、+12 V 未満かつ -12 V よりも大きく		
ケーブル長(メートル)	100 m、シールドあ	100 m、シールドありツイストペア		
デジタル出力				
出力数	4			
タイプ	リレー、ドライ接点		ソリッドステート - MOSFET	
電圧範囲	DC 5~30 V または	AC 5∼250 V	DC 20.4~28.8 V	
最大電流でロジック 1 信号			DC 20 V (最小)	
10 K Ω 負荷付きのロジック 0 信 号	DC 0.1 V (最大)			
電流(最大)	2.0 A		0.5 A	
ランプ負荷	DC 30 W / AC 200 W 5 W			
ON ステート抵抗	新規のとき最大 0.2	Ω	最大 0.6 Ω	
ポイント当たりの漏れ電流			最大 10 µA	

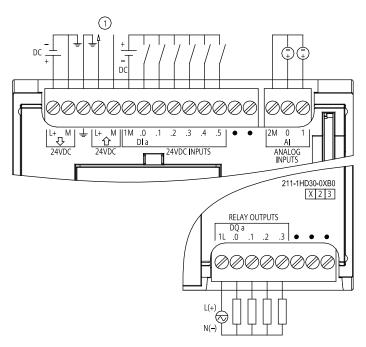
技術仕様			
モデル	CPU 1211C AC/DC/リレー	CPU 1211C DC/DC/ リレー	CPU 1211C DC/DC/DC
サージ電流	接点が閉じた状態で7	A	最大 100 ms で 8 A
過負荷保護	なし		
絶縁(フィールド側からロジック)	AC 1500 V / 1 分間(コーなし(コイルからロジッ	,	AC 500 V / 1 分間
	新規の場合 100 MΩ 最		
開いた接点間の絶縁	AC 750 V / 1 分間		
絶縁グループ	1		1
誘導クランプ電圧			L± DC 48 V、1 W 損失
切り替え遅延(Qa.0 から Qa.3)	最大 10 ms		最大 1.0 µs、オフから オン 最大 3.0 µs、オンから オフ
パルストレイン出力速度 (Qa.0 と Qa.2)	推奨しません		最大 100 KHz、 最小 2 Hz
機械寿命(負荷なし)	10,000,000 回の開閉サ	イクル	
定格負荷での接点寿命	100,000 回の開閉サイ	クル	
RUN から STOP の動作	最後の値または代替値	(デフォルト値 0)	
同時出力数	4		
ケーブル長(メートル)	500m シールドあり、1	50m シールドなし	_

配線図



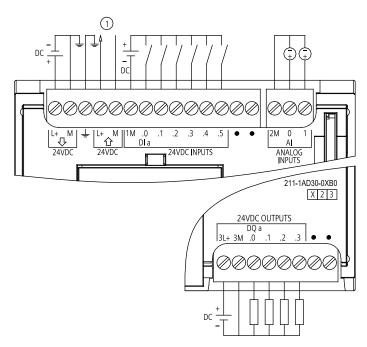
① DC 24 V センサ電源出力

図 A-1 CPU 1211C AC/DC/リレー(6ES7 211-1BD30-0XB0)



① DC 24 V センサ電源出力

図 A-2 CPU 1211C DC/DC/リレー(6ES7 211-1HD30-0XB0)



① DC 24 V センサ電源出力

図 A-3 CPU 1211C DC/DC/DC (6ES7 211-1AD30-0XB0)

A.2.2 CPU 1212C仕様

技術仕様			
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC
注文番号(MLFB)	6ES7 212-1BD30- 0XB0	6ES7 212-1HD30- 0XB0	6ES7 212-1AD30- 0XB0
全般			
外形寸法 W x H x D (mm)	90 x 100 x 75		
重量	425 グラム	385 グラム	370 グラム
電力損失	11 W 9 W		
使用可能電流(SM および CM バス)	最大 1000 mA(DC 5 V)		
使用可能電流(DC 24 V)	最大 300 mA(センサ電源)		
デジタル入力消費電流(DC 24 V)	4 mA /入力	,	

技術仕様				
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC	
CPU の機能				
ユーザーメモリ	25 キロバイト作業 保持メモリ	25 キロバイト作業メモリ / 1 メガバイトロードメモリ / 2 キロバイト保持メモリ		
オンボードデジタル I/O	8入力/6出力			
オンボードアナログ 1/0	2 入力			
プロセスイメージサイズ	入力 1024 バイト(l) /出力 1024 バイト(Q)		
ビットメモリ(M)	4096 バイト			
拡張モジュール数	最大 2 SM			
拡張ボード数	最大 1 SB			
通信モジュール拡張	最大 3 CM			
高速カウンタ	合計 4 単相: クロック速度 100 kHz × 3、30 kHz × 1 差動: クロック速度 80 kHz × 3、20 kHz × 1			
パルス出力	2			
パルスキャッチ入力	8			
時間遅延/周期割り込み	1 ms 解像度で合計	- 4		
エッジ割り込み	8 立ち上がりおよび 8 立ち下がり(12 およびオプションのシグナルボードで 12)			
メモリカード	SIMATIC メモリカ	ード(オプション)		
リアルタイムクロックの精度	+/- 60 秒/月			
リアルタイムクロック保持時間	40°C で通常 10 日	/最小6日(メンテナンス	フリーの大容量キャパシタ)	
パフォーマンス				
ビット演算実行速度	0.1 μs / 命令			
Move Word 実行速度	12 μs / 命令			
実数演算実行速度	18 µs / 命令			
通信				
ポートの数	1			
タイプ	イーサネット	イーサネット		

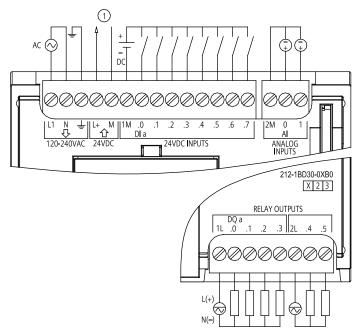
技術仕様				
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC	
接続	• HMI × 3			
	• プログラミング装置	• プログラミング装置×1		
	• ユーザープログラム	のイーサネット命令 × t	8	
	• CPU 間×3			
データ速度	10/100 Mb/秒			
絶縁(PLC ロジックへの外部信 号)	絶縁済み変圧器、DC 1	500 V		
ケーブルタイプ	CAT5e シールドあり			
電源				
電圧範囲	AC 85∼264 V	DC 20.4~28.8 V		
回線周波数	47 Hz∼63 Hz			
入力電流 CPU が最大負荷の場合のみ	AC 120 V / 80 mA AC 240 V / 40 mA	DC 24 V / 400 mA		
CPU がすべての拡張付属品を 装備し、最大負荷の場合	AC 120 V / 240 mA AC 240 V / 120 mA	DC 24 V / 1200 mA		
突入電流(最大)	AC 264 V / 20 A	DC 28.8 V / 12 A		
絶縁(入力電源からロジック)	AC 1500 V	絶縁なし		
漏電、AC ラインから機能接地	最大 0.5 mA	-		
持続時間(停電時)	AC 120 V / 20 ms AC 240 V / 80 ms	DC 24 V / 10 ms		
内部ヒューズ、ユーザー交換不 可	3 A、250 V、スローブ	ロー		
センサ電源				
電圧範囲	DC 20.4~28.8 V	L± DC 4 V (最小)		
出力電流定格(最大)	300 mA (短絡保護)			
最大リップルノイズ(10 MHz 未 満)	1 V 未満ピークツーピ ーク	入力ラインと同じ		
絶縁(CPU ロジックからセンサ 電源)	絶縁なし			

技術仕様			
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC
デジタル入力			
入力数	8		
タイプ	シンク/ソース(IEC タ	イプ 1 シンク)	
定格電圧	DC 24 V / 4 mA、(公利	尔)	
常時許容電圧	DC 30 V (最大)		
サージ電圧	DC 35 V / 0.5 秒		
ロジック 1 信号(最小)	DC 15 V / 2.5 mA		
ロジック 0 信号(最大)	DC 5 V / 1 mA		
絶縁(フィールド側からロジッ ク)	AC 500 V / 1 分間		
絶縁グループ	1		
フィルタ時間	0.2、0.4、0.8、1.6、3.2、6.4、12.8 ms (4 グループで選択可能)		
HSC クロック入力速度(最大) (ロジック 1 レベル = DC 15~ 26 V)	単相: 100 KHz (la.0~la.5)および 30 KHz (la.6~la.7) 差動: 80 KHz (la.0~la.5)および 20 KHz (la.6~la.7)		
同時入力数	8		
ケーブル長(メートル)	500m シールドあり、300m シールドなし、50m は HSC 入力用にシールドあり		
アナログ入力			
入力数	2		
タイプ	電圧(シングルエンド)		
範囲	0∼10 V		
フルスケールレンジ(データワー ド)	0~27648 (「電圧のアナログ入力表現」を参照)		
オーバーシュート範囲(データワ ード)	27,649~32,511 (「電	- 圧のアナログ入力表:	現」を参照)
オーバーフロー(データワード)	32,512~32767 (「電圧のアナログ入力表現」を参照)		
解像度	10 ビット		
最大耐電圧	DC 35 V		

技術仕様			
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC
平滑化	なし、弱、中、また 応答時間」を参照)	こは強(ステップ応答時	間については「アナログ入力
ノイズ除去	10、50、または 60 答時間」を参照)) Hz (サンプルレートに	ついては「アナログ入力応
インピーダンス	100 ΚΩ 以上		
絶縁(フィールド側からロジッ ク)	なし		
精度(25°C / 0~55°C)	フルスケールの 3.0	0% /3.5%	
コモンモード除去	40 dB、DC∼60 Hz	2	
動作信号範囲	信号およびコモンコなければなりません		
ケーブル長(メートル)	100 m のツイスト‡	およびシールドあり	
デジタル出力			
出力数	6		
タイプ	リレー、ドライ接点	Ħ.	ソリッドステート - MOSFET
電圧範囲	DC 5~30 V または	: AC 5∼250 V	DC 20.4~28.8 V
最大電流でロジック 1 信号			DC 20 V (最小)
10 K Ω 負荷付きのロジック 0 信 号			DC 0.1 V (最大)
電流(最大)	2.0 A		0.5 A
ランプ負荷	DC 30 W / AC 200	W	5 W
ON ステート抵抗	新規のとき最大 0.2	2 Ω	最大 0.6 Ω
ポイント当たりの漏れ電流			最大 10 μA
サージ電流	接点が閉じた状態で	€7A	最大 100 ms で 8 A
過負荷保護	なし		
絶縁(フィールド側からロジッ ク)	AC 1500 V / 1 分間 なし(コイルからロ	`	AC 500 V / 1 分間
絶縁抵抗	新規の場合 100 MΩ	2 最小	

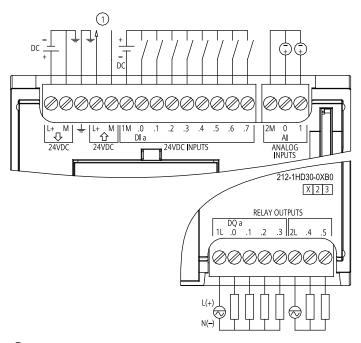
技術仕様			
モデル	CPU 1212C AC/DC/リレー	CPU 1212C DC/DC/リレー	CPU 1212C DC/DC/DC
開いた接点間の絶縁	AC 750 V / 1 分間		
絶縁グループ	2		1
誘導クランプ電圧			L± DC 48 V、1 W 損失
切り替え遅延(Qa.0 から Qa.3)	最大 10 ms		最大 1.0 µs、オフから オン 最大 3.0 µs、オンから オフ
切り替え遅延(Qa.4 から Qa.5)	最大 10 ms		最大 50 μs、オフから オン 最大 200 μs、オンから オフ
パルストレイン出力速度 (Qa.0 と Qa.2)	推奨しません		最大 100 KHz、 最小 2 Hz
機械寿命(負荷なし)	10,000,000 回の開閉	10,000,000 回の開閉サイクル	
定格負荷での接点寿命	100,000 回の開閉サイクル		
RUN から STOP の動作	最後の値または代替	<u> 値(</u> デフォルト値 0)	
同時出力数	6		
ケーブル長(メートル)	500m シールドあり	、150m シールドなし	

配線図



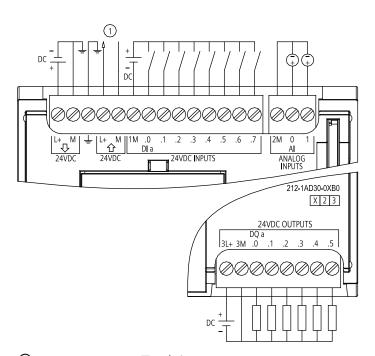
① DC 24 V センサ電源出力

図 A-4 CPU 1212C AC/DC/リレー(6ES7 212-1BD30-0XB0)



① DC 24 V センサ電源出力

図 A-5 CPU 1212C DC/DC/リレー(6ES7 212-1HD30-0XB0)



① DC 24 V センサ電源出力

図 A-6 CPU 1212C DC/DC/DC (6ES7 212-1AD30-0XB0)

A.2.3 CPU 1214C仕様

技術仕様					
モデル	CPU 1214C AC/DC/リレー	CPU 1214C DC/DC/リレー	CPU 1214C DC/DC/DC		
注文番号(MLFB)	6ES7 214-1BE30-	6ES7 214-1HE30-	6ES7 214-1AE30-		
	0XB0	0XB0	0XB0		
全般					
外形寸法 W x H x D (mm)	110 x 100 x 75				
重量	475 グラム	435 グラム	415 グラム		
電力損失	14 W	12 W			
使用可能電流(SM および CM バ	最大 1600 mA (DC 5 \	V)			
ス)					
使用可能電流(DC 24 V)	最大 400 mA(センサ電	這源)			
デジタル入力消費電流(DC 24	4 mA/入力				
V)					
CPU の機能					
ユーザーメモリ	50 キロバイト作業メモリ / 2 メガバイトロードメモリ / 2 キロバイト保持メモリ				
オンボードデジタル I/O	14 入力 / 10 出力				
オンボードアナログ 1/0	2 入力				
プロセスイメージサイズ	入力 1024 バイト(I) /ヒ	出力 1024 バイト(Q)			
ビットメモリ(M)	8192 バイト				
拡張モジュール数	最大 8 SM				
拡張ボード数	最大 1 SB				
通信モジュール拡張	最大 3 CM				
高速カウンタ	合計 6				
	単相: クロック速度 10	00 kHz × 3、30 kHz × 3	}		
	差動: クロック速度 80 kHz × 3、20 kHz × 3				
パルス出力	2				
パルスキャッチ入力	14				
時間遅延/周期割り込み	1 ms 解像度で合計 4				
エッジ割り込み	12 立ち上がりおよび 12 立ち下がり(14 およびオプションのシグナルボードで 14)				
メモリカード	SIMATIC メモリカード(オプション)				

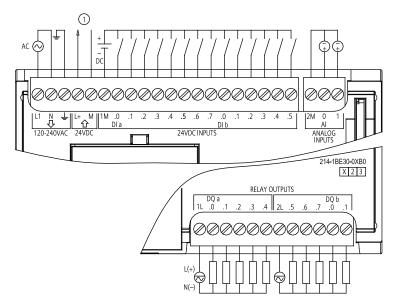
技術仕様	T		
モデル	CPU 1214C AC/DC/リレー	CPU 1214C DC/DC/リレー	CPU 1214C DC/DC/DC
リアルタイムクロックの精度	+/- 60 秒/月		
リアルタイムクロック保持時間	40°C で通常 10 日/最	小6日(メンテナンス)	フリーの大容量キャパシタ
パフォーマンス			
ビット演算実行速度	0.1 μs / 命令		
Move Word 実行速度	12 μs / 命令		
実数演算実行速度	18 μs / 命令		
ポートの数	1		
タイプ	イーサネット		
接続	• HMI × 3		
	• プログラミング装置	置×1	
	• ユーザープログラ、	ムのイーサネット命令	`×8
	• CPU 間 × 3		
データ速度	10/100 Mb/秒		
絶縁(PLC ロジックへの外部信 号)	絶縁済み変圧器、DC	1500 V	
ケーブルタイプ	CAT5e シールドあり		
電源			
電圧範囲	AC 85∼264 V	DC 20.4~28.8 V	
回線周波数	47 Hz∼63 Hz		
入力電流 CPU が最大負荷の場合のみ	AC 120 V / 100 mA	DC 24 V / 500 mA	
	AC 240 V / 50 mA		
CPU がすべての拡張付属品を	AC 120 V / 300 mA	DC 24 V / 1500 mA	1
装備し、最大負荷の場合	AC 240 V / 150 mA		
突入電流(最大)	AC 264 V / 20 A	DC 28.8 V / 12 A	
絶縁(入力電源からロジック)	AC 1500 V	絶縁なし	
漏電、AC ラインから機能接地	最大 0.5 mA	-	
持続時間(停電時)	AC 120 V / 20 ms AC 240 V / 80 ms	DC 24 V / 10 ms	
内部ヒューズ、ユーザー交換不 可	3 A、250 V、スローフ		

技術仕様			
モデル	CPU 1214C AC/DC/リレー	CPU 1214C DC/DC/リレー	CPU 1214C DC/DC/DC
センサ電源			
電圧範囲	DC 20.4~28.8 V	L± DC 4 V (最小)	
出力電流定格(最大)	400 mA (短絡保護)		
最大リップルノイズ(10 MHz 未 満)	1 V 未満ピークツーピ ーク	入力ラインと同じ	
絶縁(CPU ロジックからセンサ 電源)	絶縁なし		
デジタル入力			
入力数	14		
タイプ	シンク/ソース(IEC タイ	`プ 1 シンク)	
定格電圧	DC 24 V / 4 mA、(公称	5)	
常時許容電圧	DC 30 V (最大)		
サージ電圧	DC 35 V / 0.5 秒		
ロジック 1 信号(最小)	DC 15 V / 2.5 mA		
ロジック 0 信号(最大)	DC 5 V / 1 mA		
絶縁(フィールド側からロジッ ク)	AC 500 V / 1 分間		
絶縁グループ	1		
フィルタ時間	0.2、0.4、0.8、1.6、3	.2、6.4、12.8 ms (4 グ	ループで選択可能)
HSC クロック入力速度(最大) (ロジック 1 レベル = DC 15~ 26 V)	単相: 100 KHz (la.0~la 差動: 80 KHz (la.0~la.	,	,
同時入力数	14		
ケーブル長(メートル)	500m シールドあり、3 ルドあり	300m シールドなし、50	Om は HSC 入力用にシー
アナログ入力			
入力数	2		
タイプ	電圧(シングルエンド)		
範囲	0∼10 V		
フルスケールレンジ (データワー ド)	0~27648 (「電圧のア	ナログ入力表現」を参照	照)

技術仕様					
モデル	CPU 1214C AC/DC/リレー	CPU 1214C DC/DC/リレー	CPU 1214C DC/DC/DC		
オーバーシュート範囲(データワ	27,649~32,511 (「電圧のアナログ入力表	現」を参照)		
- ₹)					
オーバーフロー(データワード)	32,512~32767 (電圧のアナログ入力表現	現」を参照)		
解像度	10 ビット				
最大耐電圧	DC 35 V				
平滑化	なし、弱、中、ま7 応答時間」を参照)	たは強(ステップ応答時間	間については「アナログ入力		
ノイズ除去	10、50、または 60 答時間」を参照)) Hz (サンプルレートに	ついては「アナログ入力応		
インピーダンス	100 KΩ 以上				
絶縁(フィールド側からロジッ ク)	なし				
精度(25°C / 0∼55°C)	フルスケールの 3.0)% /3.5%			
コモンモード除去	40 dB、DC∼60 H	7			
動作信号範囲	信号およびコモン	モード電圧は、 +12 V 未	満かつ -12 V よりも大きく		
	なければなりません	\mathcal{V}_{\circ}			
ケーブル長(メートル)	100 m のツイスト	およびシールドあり			
デジタル出力	,				
出力数	10				
タイプ	リレー、ドライ接点	点	ソリッドステート -		
最 了放回	DO 5 00 1/ 3-3-11	- 10.5050.\/	MOSFET		
電圧範囲	DC 5~30 V または	. AC 5~250 V	DC 20.4~28.8 V		
最大電流でロジック 1 信号			DC 20 V (最小)		
10 K Ω 負荷付きのロジック 0 信 号	DC 0.1 V (最大)				
電流(最大)	2.0 A 0.5 A				
ランプ負荷	DC 30 W / AC 200 W 5 W				
ON ステート抵抗	新規のとき最大 0.2	2 Ω	最大 0.6 Ω		
ポイント当たりの漏れ電流	最大 10 μA				
サージ電流	接点が閉じた状態で 7 A 最大 100 ms で 8 A				
過負荷保護	なし				

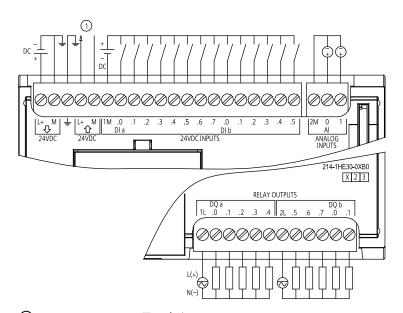
技術仕様			
モデル	CPU 1214C AC/DC/リレー	CPU 1214C DC/DC/リレー	CPU 1214C DC/DC/DC
絶縁(フィールド側からロジック)	AC 1500 V / 1 分間なし(コイルからロ		AC 500 V / 1 分間
絶縁抵抗	新規の場合 100 MG	0 最小	
開いた接点間の絶縁	AC 750 V / 1 分間		
絶縁グループ	2		1
誘導クランプ電圧			L± DC 48 V、1 W 損失
切り替え遅延(Qa.0 から Qa.3)	最大 10 ms		最大 1.0 µs、オフから オン 最大 3.0 µs、オンから オフ
切り替え遅延(Qa.4 から Qa.1)	最大 10 ms		最大 50 µs、オフから オン 最大 200 µs、オンから オフ
パルストレイン出力速度 (Qa.0 と Qa.2)	推奨しません		最大 100 KHz、 最小 2 Hz
機械寿命(負荷なし)	10,000,000 回の開	閉サイクル	
定格負荷での接点寿命	100,000 回の開閉サイクル		
RUN から STOP の動作	最後の値または代替	 	
同時出力数	10		
ケーブル長(メートル)	500m シールドあり)、150m シールドなし	

配線図



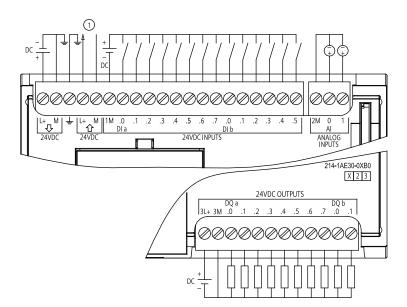
① DC 24 V センサ電源出力

図 A-7 CPU 1214C AC/DC/リレー(6ES7 214-1BE30-0XB0)



① DC 24 V センサ電源出力

図 A-8 CPU 1214C DC/DC/リレー(6ES7 214-1HE30-0XB0)



① DC 24 V センサ電源出力

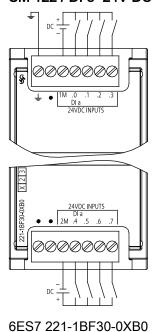
図 A-9 CPU 1214C DC/DC/DC (6ES7 214-1AE30-0XB0)

A.3.1 SM 1221 デジタル入力仕様

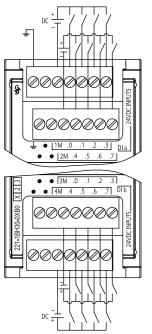
技術仕様		
モデル	SM 1221 DI 8×24V DC	SM 1221 DI 16×24V DC
注文番号(MLFB)	6ES7 221-1BF30-0XB0	6ES7 221-1BH30-0XB0
全般		
外形寸法 W x H x D (mm)	45 x 100 x 75	
重量	170 グラム	210 グラム
電力損失	1.5 W	2.5 W
消費電流(SM バス)	105 mA	130 mA
消費電流(DC 24 V)	4 mA /入力	4 mA /入力
デジタル入力		
入力数	8	16
タイプ	シンク/ソース(IEC タイプ 1 シンク	")
定格電圧	DC 24 V / 4 mA、(公称)	
常時許容電圧	DC 30 V (最大)	
サージ電圧	DC 35 V / 0.5 秒	
ロジック 1 信号(最小)	DC 15 V / 2.5 mA	
ロジック 0 信号(最大)	DC 5 V / 1 mA	
絶縁(フィールド側からロジッ	AC 500 V / 1 分間	
ク)		
絶縁グループ	2	4
フィルタ時間	0.2、0.4、0.8、1.6、3.2、6.4、12	2.8 ms (4 グループで選択可能)
同時入力数	8	16
ケーブル長(メートル)	500m シールドあり、300m シール	ドなし

配線図

SM 1221 DI 8×24V DC



SM 1221 DI 16×24V DC



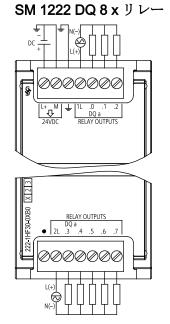
6ES7 221-1BH30-0XB0

A.3.2 SM 1222 デジタル出力仕様

技術仕様				
モデル	SM 1222 DQ 8x リレー	SM1222 DQ 16x リレー	SM1222 DQ 8×24V DC	SM1222 DQ 16×24V DC
注文番号(MLFB)	6ES7 222-	6ES7 222-	6ES7 222-	6ES7 222-
	1HF30-0XB0	1HH30-0XB0	1BF30-0XB0	1BH30-0XB0
全般				
外形寸法 W x H x D (mm)	45 x 100 x 75			
重量	190 グラム	260 グラム	180 グラム	220 グラム
電力損失	4.5 W	8.5 W	1.5 W	2.5 W
消費電流(SM バス)	120 mA	135 mA	120 mA	140 mA
消費電流(DC 24 V)	11 mA /リレーコ	コイル使用		

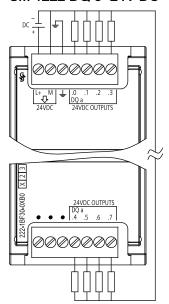
技術仕様						
モデル	SM 1222 DQ 8x リレー	SM1222 DQ 16x リレー	SM1222 DQ 8×24V DC	SM1222 DQ 16×24V DC		
デジタル出力						
出力数	8	16	8	16		
タイプ	リレー、ドライ技		ソリッドステー	ト - MOSFET		
電圧範囲	DC 5~30 V また	はAC 5~250 V	DC 20.4~28.8 V	/		
最大電流でロジック 1 信号			DC 20 V (最小)			
10 K Ω 負荷付きのロジック 0 信号			DC 0.1 V (最大)			
電流(最大)	2.0 A		0.5 A			
ランプ負荷	DC 30 W / AC 20	00 W	5W			
ON ステート接点抵抗	新規のとき最大().2 Ω	最大 0.6 Ω			
ポイント当たりの漏れ電流			最大 10 μA			
サージ電流	接点が閉じた状態	Eで7A	最大 100 ms で 8 A			
過負荷保護	なし					
絶縁(フィールド側からロジッ	AC 1500 V / 1 分	間(コイルから接	AC 500 V / 1 分間			
ク)	点)					
	なし(コイルから	ロジック)				
絶縁抵抗	新規の場合 100 M	ΜΩ 最小				
開いた接点間の絶縁	AC 750 V / 1 分間	<u> </u>				
絶縁グループ	2	4	1	1		
共通定格による電流(最大.)	10 A		4 A	8 A		
誘導クランプ電圧			L±48 V、1 W 損	失		
切り替え遅延	最大 10 ms		オフからオン最大	大 50 µs		
	オフからオン最大 200 μs			大 200 µs		
機械寿命(負荷なし)	10,000,000 回の	開閉サイクル				
定格負荷での接点寿命	100,000 回の開閉	ラサイクル				
RUN から STOP の動作	最後の値または代替値(デフォルト値 0)					
同時出力数	8	16	8	16		
ケーブル長(メートル) 500m シールドあり、150m シールドなし						

配線図



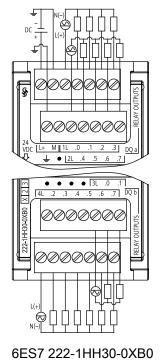
6ES7 222-1HF30-0XB0

SM 1222 DQ 8×24V DC

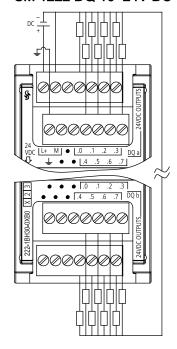


6ES7 222-1BF30-0XB0

SM 1222 DQ 16 x リレー



SM 1222 DQ 16×24V DC



6ES7 222-1BH30-0XB0

A.3.3 SM 1223 デジタル入出力仕様

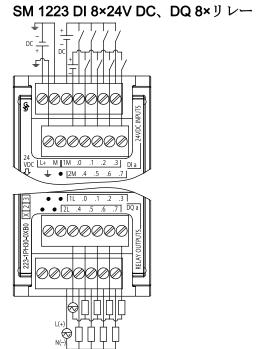
技術仕様				
モデル	SM 1223 DI 8×24V DC、DQ 8×リレー	SM 1223 DI 16×24V DC、 DQ 16×リレー	SM 1223 DI 8×24V DC、DQ 8×24V DC	SM 1223 DI 16×24V DC、 DQ16×24V DC
注文番号(MLFB)	6ES7 223- 1PH30-0XB0	6ES7 223- 1PL30-0XB0	6ES7 223- 1BH30-0XB0	6ES7 223- 1BL30-0XB0
外形寸法 W x H x D (mm)	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75	70 x 100 x 75
重量	230 グラム	350 グラム	210 グラム	310 グラム
電力損失	5.5 W	10 W	2.5 W	4.5 W
消費電流(SM バス)	145 mA	180 mA	145 mA	185 mA
消費電流(DC 24 V)	4 mA /入力 11 mA /リレーコイル使用		4 mA /入力	

技術仕様							
モデル	SM 1223 DI 8×24V DC、DQ 8×リレー	SM 1223 DI 16×24V DC、 DQ 16×リレー	SM 1223 DI 8×24V DC、DQ 8×24V DC	SM 1223 DI 16×24V DC、 DQ16×24V DC			
デジタル入力	デジタル入力						
入力数	8	16	8	16			
タイプ	シンク/ソース(IE	Cタイプ 1 シンク)				
定格電圧	DC 24 V / 4 mA、	(公称)					
常時許容電圧	DC 30 V (最大)						
サージ電圧	DC 35 V / 0.5 秒						
ロジック 1 信号(最小)	DC 15 V / 2.5 mA	1					
ロジック 0 信号(最大)	DC 5 V / 1 mA						
絶縁(フィールド側からロジッ ク)	AC 500 V / 1 分間						
絶縁グループ	2	2	2	2			
フィルタ時間	0.2、0.4、0.8、1	.6、3.2、6.4、12	2.8 ms (4 グループ	で選択可能)			
同時入力数	8	16	8	16			
ケーブル長(メートル)	500m シールドあ	り、300m シール	ドなし				
デジタル出力							
出力数	8	16	8	16			
タイプ	リレー、ドライ接		ソリッドステート - MOSFET				
電圧範囲	DC 5~30 V また	は AC 5~250 V	DC 20.4~28.8 V				
最大電流でロジック 1 信号			DC 20 V (最小)				
10 K Ω 負荷付きのロジック 0 信号	DC 0.1 V (最大)						
電流(最大)	2.0 A		0.5 A				
ランプ負荷	DC 30 W / AC 200 W		5 W				
ON ステート接点抵抗	新規のとき最大 0.2 Ω 最大 0.6 Ω						
ポイント当たりの漏れ電流	最大 10 μA						
サージ電流	接点が閉じた状態	で7A	最大 100 ms で 8 A				
過負荷保護	なし						

技術仕様					
モデル	SM 1223 DI 8×24V DC、DQ 8×リレー	SM 1223 DI 16×24V DC、 DQ 16×リレー	SM 1223 DI 8×24V DC、DQ 8×24V DC	SM 1223 DI 16×24V DC DQ16×24V DC	
絶縁(フィールド側からロジック)	AC 1500 V / 1 分間(コイルから接点) なし(コイルからロジック)		接 AC 500 V / 1 分間		
絶縁抵抗	新規の場合 100 N	//Ω 最小			
開いた接点間の絶縁	AC 750 V / 1 分間				
絶縁グループ	2	4	1	1	
共通定格による電流	10A	8 A	4 A	8 A	
誘導クランプ電圧			L±48 V、1 W 損失		
切り替え遅延	最大 10 ms		最大 50 µs、オフ 最大 200 µs、オン		
機械寿命(負荷なし)	10,000,000 回の間	開閉サイクル			
定格負荷での接点寿命	100,000 回の開閉サイクル				
RUN から STOP の動作	最後の値または代替値(デフォルト値 0)				
同時出力数	8	16	8	16	
ケーブル長(メートル)	500m シールドあり、150m シールドなし				

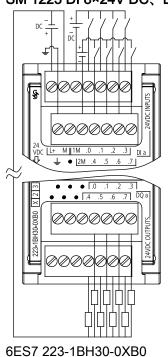
配線図



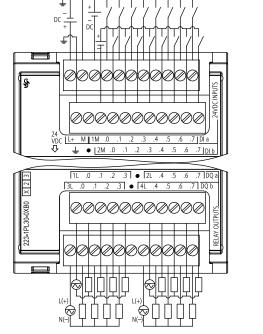


6ES7 223-1PH30-0XB0

SM 1223 DI 8×24V DC、DQ 8×24V DC

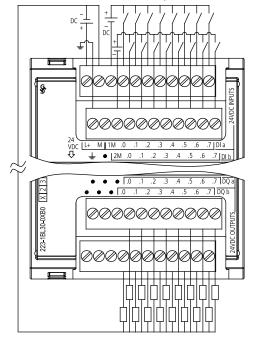


SM1223 DI 16×24V DC、DQ 16×リレー



6ES7 223-1PL30-0XB0

SM 1223 DI 16×24V DC、DQ 16×24V DC



6ES7 223-1BL30-0XB0

A.4.1 SM 1231、SM 1232、SM 1234 アナログ仕様

技術仕様					
モデル	SM 1231 Al 4x13 ビ ット	SM 1231 Al 8x13 ビット	SM 1234 AI 4x13 ビット AQ 2x14 ビット		
注文番号(MLFB)	6ES7 231-4HD30- 0XB0	6ES7 231-4HF30- 0XB0	6ES7 234-4HE30- 0XB0		
全般			•		
外形寸法 W x H x D (mm)	45 x 100 x 75	45 x 100 x 75	45 x 100 x 75		
重量	180 グラム	180 グラム	220 グラム		
電力損失	1.5 W	1.5 W	2.0 W		
消費電流(SM バス)	80 mA	90 mA	80 mA		
消費電流(DC 24 V)	45 mA	45 mA	60 mA (負荷なし)		
アナログ入力					
入力数	4	8	4		
タイプ	電圧または電流(差動)	: 2 つのグループから選	択可能		
範囲	±10 V、±5 V、±2.5 V	、または 0~20 mA			
フルスケールレンジ(データワ ード)	-27,648~27,648				
オーバーシュート範囲	電圧: 32,511~27,649	/ -27,649~-32,512			
(データワード)	電流: 32,511~27,649	/ 0∼-4864			
	(「電圧のアナログ入」 ジ 427) を参照)	力表現」、「電流のアナ	ログ入力表現」 (ペー		
オーバーフロー/アンダーフロ	電圧: 32,767~32,512	/ -32,513~-32,768			
ー(データワード)	電流: 32,767~32,512	/ -4865~-32,768			
	(「電圧のアナログ入力表現」、「電流のアナログ入力表現」 (ページ 427)を参照)				
解像度	12 ビット+符号ビット				
最大耐電圧/電流	±35 V / ±40 mA	±35 V / ±40 mA			
平滑化		なし、弱、中、または強(ステップ応答時間については「アナログ入力 応答時間」(ページ 427)を参照)			

技術仕様					
モデル	SM 1231 AI 4x13 ビット	SM 1231 AI 8x13 ビット	SM 1234 AI 4x13 ビッ ト AQ 2x14 ビット		
ノイズ除去	400、60、50、または 力応答時間」 (ページ	•	については 「アナログ入		
インピーダンス	≥ 9 MΩ (電圧) / 250 Ω	(電流)			
絶縁(フィールド側からロジック)	なし				
精度(25°C / 0~55°C)	フルスケールの±0.1%	/ ±0.2%			
アナログからデジタルへの変換 時間	625 μs (400 Hz 除去)				
コモンモード除去	40 dB、DC∼60 Hz				
動作信号範囲	信号およびコモンモード電圧は、+12 V 未満かつ -12 V よりも大きく なければなりません。				
ケーブル長(メートル)	100m ツイストおよびシールドあり				
診断					
オーバーフロー/アンダーフロ	あり1	あり1	あり1		
短絡接地(電圧モードのみ)	適用なし	適用なし	出力に適用		
断線(電流モードのみ)	適用なし	適用なし	出力に適用		
DC 24 V (低電圧)	あり あり あり				

¹ DC +30 V を超えるか、または DC -15 V 未満の電圧が入力に適用されると、結果の値は不明になり、対応 するオーバーフローまたはアンダーフローが有効にならない場合があります。

技術仕様			
モデル	SM 1232 AQ 2x14 ビット	SM 1232 AQ 4x14 ビット	SM 1234 AI 4x13 ビッ ト AQ 2x14 ビット
注文番号(MLFB)	6ES7 232-4HB30- 0XB0	6ES7 232-4HD30- 0XB0	6ES7 234-4HE30- 0XB0
全般	•		
外形寸法 W x H x D (mm)	45 x 100 x 75	45 x 100 x 75	45 x 100 x 75
重量	180 グラム	180 グラム	220 グラム
電力損失	1.5 W	1.5 W	2.0 W
消費電流(SM バス)	80 mA	80 mA	80 mA
消費電流(DC 24 V)	45 mA (負荷なし)	45 mA (負荷なし)	60 mA (負荷なし)

技術仕様						
モデル	SM 1232 AQ 2x14 ビット	SM 1232 AQ 4x14 ビット	SM 1234 AI 4x13 ビット AQ 2x14 ビット			
アナログ出力						
出力数	2	4	2			
タイプ	電圧または電流					
範囲	±10 V または 0~20 m	A				
解像度	電圧: 14 ビット、電流	: 13 ビット				
フルスケールレンジ (データワード)	電圧: -27,648~27,648、電流: 0~27,648 (「電圧のアナログ出力表現」と「電流のアナログ出力表現」 (ページ 427)を参照)					
精度(25°C / 0~55°C)	フルスケールの±0.3%	/ ±0.6%				
設定時間(新規値の 95%)	電圧: 300 μS (R)、750 μS (1 uF)。電流: 600 μS (1 mH)、2 ms (10 mH)					
負荷インピーダンス	電圧: 1000 Ω 以上、電流: 600 Ω 以下					
RUN から STOP の動作	最後の値または代替値(デフォルト値 0)					
絶縁(フィールド側からロジック)	なし					
ケーブル長(メートル)	100m ツイストおよびシールドあり					
診断						
オーバーフロー/アンダーフロー	あり	あり	あり 1			
短絡接地(電圧モードのみ)	あり	あり	出力に適用			
断線(電流モードのみ)	あり	あり	出力に適用			
DC 24 V (低電圧)	あり あり あり					

¹ DC +30 V を超えるか、または DC -15 V 未満の電圧が入力に適用されると、結果の値は不明になり、対応するオーバーフローまたはアンダーフローが有効にならない場合があります。

アナログ入力応答時間

SM アナログモジュールステップ応答(ms)					
	95% で	:测定 0 V∼10 V	,		
平滑化選択		遮断	周波数		
	400 Hz 60 Hz 50 Hz 10 H				
なし	4	18	22	100	
弱	9	52	63	320	
中	32	203	241	1200	
強	61	400	483	2410	
サンプルレート					
サンプルレート • 4 チャンネル • 8 チャンネル	• 0.625	• 4.17	• 5	• 25	
• 8チャンネル	• 1.25	• 4.17	• 5	• 25	

CPU アナログ入力ステップ応答(ms)					
	95%で測定 0 V~10 V				
平滑化選択	遮断周波数				
	60 Hz 50 Hz 10 Hz				
なし	63	65	130		
弱	84	93	340		
中	221	258	1210		
強	424	499	2410		
サンプルレート	4.17	5	25		

電圧のアナログ入力表現

システム	電圧測定範囲							
10 進数	16 進数	± 10 V	±5 V	±2.5 V	±2.5 V		0~10 V	
32767	7FFF	11.851 V	5.926 V	2.963 V	オーバーフロー	11.851V	オーバーフ	
32512	7F00						ロー	
32511	7EFF	11.759 V	5.879 V	2.940 V	オーバーシュー	11.759 V	オーバーシ	
27649	6C01				卜範囲		ュート範囲	
27648	6C00	10 V	5 V	2.5 V	定格範囲	10 V	定格範囲	
20736	5100	7.5 V	3.75 V	1.875 V		7.5 V		
1	1	361.7 μV	180.8 μV	90.4 μV		361.7 μV		
0	0	0 V	0 V	0 V		0 V		
-1	FFFF					負の値はサ		
-20736	AF00	-7.5 V	-3.75 V	-1.875 V		ポートされ		
-27648	9400	-10 V	-5 V	-2.5 V		ていません。		
-27649	93FF				アンダーシュー	700		
-32512	8100	-11.759 V	-5.879 V	-2.940 V	卜範囲			
-32513	80FF				アンダーフロー			
-32768	8000	-11.851 V	-5.926 V	-2.963 V				

電流のアナログ入力表現

システム	電流測定範囲					
10 進数	16 進数	0 mA~20 mA				
32767	7FFF	23.70 mA	オーバーフロー			
32512	7F00					
32511	7EFF	23.52 mA	オーバーシュート範囲			
27649	6C01					
27648	6C00	20 mA	定格範囲			
20736	5100	15 mA				
1	1	723.4 nA				
0	0	0 mA				
-1	FFFF		アンダーシュート範囲			
-4864	ED00	-3.52 mA				
-4865	ECFF		アンダーフロー			
-32768	8000					

電圧のアナログ出力表現

システム			電圧出力範囲
10 進数	16 進数	± 10 V	
32767	7FFF	注1を参照	オーバーフロー
32512	7F00	注1を参照	
32511	7EFF	11.76 V	オーバーシュート範囲
27649	6C01		
27648	6C00	10 V	定格範囲
20736	5100	7.5 V	
1	1	361.7 μV	
0	0	0 V	
-1	FFFF	-361.7 μV	
-20736	AF00	-7.5 V	
-27648	9400	-10 V	
-27649	93FF		アンダーシュート範囲
-32512	8100	-11.76 V	
-32513	80FF	注1を参照	アンダーフロー
-32768	8000	注1を参照	

¹ オーバーフローまたはアンダーフロー条件で、アナログ出力は、アナログシグナルモジュールに設定されたデバイスコンフィグレーションプロパティに従って動作します。 [Reaction to CPU STOP]パラメータで、次のいずれかを選択します。 代替値を使用するか、最後の値を保持する。

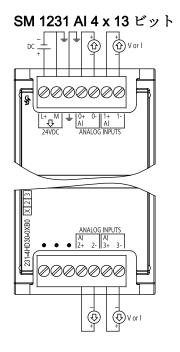
A.4 アナログシグナルモジュール(SM)

電流のアナログ出力表現

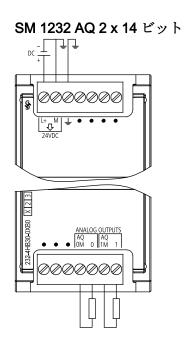
システム		電流出力範囲		
10 進数	16 進数	± 20 mA		
32767	7FFF	注1を参照	オーバーフロー	
32512	7F00	注 1 を参照		
32511	7EFF	23.52 mA	オーバーシュート範囲	
27649	6C01			
27648	6C00	20 mA	定格範囲	
20736	5100	15 mA		
1	1	723.4 nA		
0	0	0 mA		
-1	FFFF		アンダーシュート範囲	
-32512	8100			
-32513	80FF	注1を参照	アンダーフロー	
-32768	8000	注1を参照		

1. オーバーフローまたはアンダーフロー条件で、アナログ出力は、アナログシグナルモジュールに設定されたデバイスコンフィグレーションプロパティに従って動作します。 [Reaction to CPU STOP]パラメータで、次のいずれかを選択します。 代替値を使用するか、最後の値を保持する。

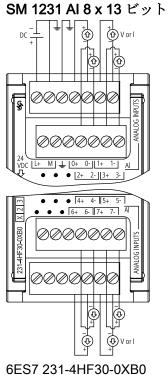
配線図

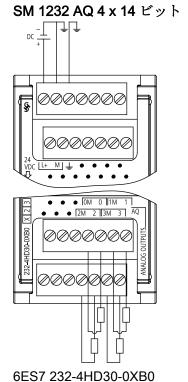


6ES7 231-4HD30-0XB0



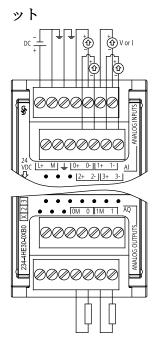
6ES7 232-4HB30-0XB0





A.4 アナログシグナルモジュール(SM)

SM 1234 Al 4 x 13 ビット / AQ 2 x 14 ビ



6ES7 234-4HE30-0XB0

A.5 シグナルボード(SB)

A.5.1 SB 1223 2×24V DC入力 / 2×24V DC出力の仕様

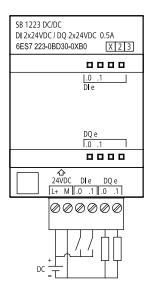
デジタルシグナルボードの仕様

技術データ				
モデル	SB 1223 DI 2×24V DC、DQ 2×24V DC			
注文番号(MLFB)	6ES7 223-0BD30-0XB0			
全般				
外形寸法 W x H x D (mm)	38 x 62 x 21			
重量	40 グラム			
電力損失	1.0 W			
消費電流(SM バス)	50 mA			
消費電流(DC 24 V)	4 mA /入力			
デジタル入力				
入力数	2			
タイプ	IEC タイプ 1 シンク			
定格電圧	DC 24 V / 4 mA、(公称)			
常時許容電圧	DC 30 V (最大)			
サージ電圧	DC 35 V / 0.5 秒			
ロジック 1 信号(最小)	DC 15 V / 2.5 mA			
ロジック 0 信号(最大)	DC 5 V / 1 mA			
HSC クロック入力レート(最大)	20 kHz (DC 15~30 V) 30 kHz (DC 15~26 V)			
絶縁(フィールド側からロジック)	AC 500 V / 1 分間			
絶縁グループ	1			
フィルタ時間	0.2、0.4、0.8、1.6、3.2、6.4、12.8 ms 2 グループで選択可能			
同時入力数	2			

A.5 シグナルボード(SB)

技術データ				
モデル	SB 1223 DI 2×24V DC、DQ 2×24V DC			
ケーブル長(メートル)	500m シールドあり、300m シールドなし			
デジタル出力				
出力数	2			
出力タイプ	ソリッドステート - MOSFET			
電圧範囲	DC 20.4~28.8 V			
最大電流でロジック 1 信号	DC 20 V (最小)			
10 K Ω 負荷付きのロジック 0 信号	DC 0.1 V (最大)			
電流(最大)	0.5 A			
ランプ負荷	5 W			
ON ステート接点抵抗	最大 0.6 Ω			
ポイント当たりの漏れ電流	最大 10 μA			
パルストレイン出力レート	最大 20 KHz、最小 2 Hz			
サージ電流	最大 100 ms で 5 A			
過負荷保護	なし			
絶縁(フィールド側からロジック)	AC 500 V / 1 分間			
絶縁グループ	1			
共通定格による電流	1 A			
誘導クランプ電圧	L±48 V、1 W 損失			
切り替え遅延	最大 2 µs、オフからオン			
	最大 10 µs、オンからオフ			
RUN から STOP の動作	最後の値または代替値(デフォルト値 0)			
同時出力数	2			
ケーブル長(メートル)	500m シールドあり、150m シールドなし			

SB 1223 2×24V DC 入力 / 2×24V DC 出力配線図



A.5.2 SB 1232 1 アナログ出力仕様

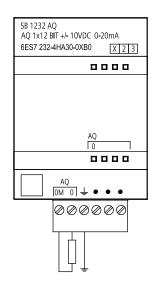
アナログシグナルボードの仕様

技術データ				
モデル	SB 1223 AQ 1x12 ビット			
注文番号(MLFB)	6ES7 232-4HA30-0XB0			
全般				
外形寸法 W x H x D (mm)	38 x 62 x 21 mm			
重量	40 グラム			
電力損失	1.5 W			
消費電流(SM バス)	15 mA			
消費電流(DC 24 V)	40 mA (負荷なし)			
アナログ出力				
出力数	1			
タイプ	電圧または電流			
範囲	±10 V または 0~20 mA			

A.5 シグナルボード(SB)

技術データ				
モデル	SB 1223 AQ 1x12 ビット			
解像度	電圧: 12 ビット 電流: 11 ビット			
フルスケールレンジ(データワード)	電圧: -27,648~27,648 電流: 0~27,648			
精度(25°C / 0~55°C)	フルスケールの±0.5% / ±1%			
設定時間(新規値の 95%)	電圧: 300 µS (R)、750 µS (1 uF) 電流: 600 µS (1 mH)、2 ms (10 mH)			
負荷インピーダンス	電圧: 1000 Ω 以上 電流: 600 Ω 以下			
RUN から STOP の動作	最後の値または代替値(デフォルト値 0)			
絶縁(フィールド側からロジック)	なし			
ケーブル長(メートル)	100 m のツイストおよびシールドあり			
診断				
オーバーフロー/アンダーフロー	あり			
短絡接地(電圧モードのみ)	あり			
断線(電流モードのみ)	あり			

SB 1232 1 x アナログ出力配線図



A.6 通信モジュール(CM)

A.6.1 CM 1241 RS485 仕様

表 A-1 通信モジュール CM 1241 RS485

技術データ	
注文番号(MLFB)	6ES7 241-1CH30-0XB0
外形寸法と重量	
寸法	30 x 100 x 75 mm
重量	150 グラム
送信側と受信側	
コモンモード電圧範囲	-7 V∼+12 V、1 秒、3 VRMS 常時
送信側差動出力電圧	R _L = 100 Ω のとき最小 2 V
	R _L = 54 Ω のとき最小 1.5 V
終端およびバイアス	Bで10KΩ~+5 V、PROFIBUSピン3
	Aで10KΩ~GND、PROFIBUSピン8
受信側入力インピーダンス	終端を含む 5.4Κ Ω 分
受信側閾値/感度	最小+/- 0.2 V、60 mV ティピカルヒステリシス
絶縁	500 VAC、1 分
RS485 信号から筐体接地	
RS485 信号から CPU 共通ロジック	
ケーブル長、シールド済み	最長 1000 m
電源仕様	
電力損失(損失)	1.1 W
DC +5 V から	220 mA

ピン	説明	1.1	コネク	タ	ピン	説明
			(メス))		
1 GND	ロジックまたは通信接地	9		5	6 PWR	+5 V、100 オーム直列レジスタ付き:
		8		4		出力
2	接続されていません	7		3	7	接続されていません
3 TxD+	信号 B (RxD/TxD+): 入力/出力	6		1	8 TXD-	信号 A (RxD/TxD-): 入力/出力
4 RTS	送信要求(TTL レベル): 出力		_		9	接続されていません
5 GND	ロジックまたは通信接地				SHELL	筐体接地

A.6 通信モジュール(CM)

A.6.2 CM 1241 RS232 仕様

通信モジュール CM 1241 RS232

技術データ	
注文番号(MLFB)	6ES7 241-1AH30-0XB0
外形寸法と重量	
寸法	30 x 100 x 75 mm
重量	150 グラム
送信側と受信側	
送信側出力電圧	R _L = 3K Ω のとき最小+/- 5 V
送信側出力電圧	DC +/- 15 V (最大)
受信側入力インピーダンス	最小3ΚΩ
受信側閾値/感度	0.8 V 最小(LOW)、2.4 V 最大(HIGH)
	0.5 V (ティピカルヒステリシス)
受信側入力電圧	DC +/- 30 V (最大)
絶縁	500 VAC、1 分
RS 232 信号から筐体接地	
RS 232 信号から CPU 共通ロジック	
ケーブル長、シールド済み	最長 10 m
電源仕様	
電力損失(損失)	1.1 W
DC +5 V から	220 mA

ピン	説明	コネクタ (オス)	ピン	説明
1 DCD	データキャリア検出: 入力	6 0 1	6 DSR	データセットレディ: 入力
2 RxD	DCE からの受信データ: 入力	7 0 0 2	7 RTS	送信要求: 出力
3 TxD	DCE に送信されるデータ: 出力	8 0 4 5	8 CTS	送信可: 入力
4 DTR	データ端末レディ: 出力		9 RI	リングインジケータ(使用しません)
5 GND	ロジックグラウンド		SHELL	筐体接地

A.7 SIMATIC メモリカード

メモリカードの仕様

注文番号	容量
6ES7 954-8LF00-0AA0	24 MB
6ES7 954-8LB00-0AA0	2 MB

A.8 入力シミュレータ

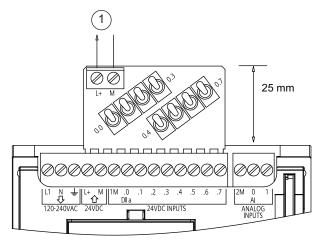
モデル	8 ポジションシミュレータ	14 ポジションシミュレー タ
注文番号(MLFB)	6ES7 274-1XF30-0XA0	6ES7 274-1XH30-0XA0
外形寸法 W x H x D (mm)	43 x 35 x 23	67 x 35 x 23
重量	20 グラム	30 グラム
ポイント	8	14
CPU と共に使用	CPU 1211C、CPU 1212C	CPU 1214C

! 警告

これらの入力シミュレータはクラス | ディビジョン 2、またはクラス | ゾーン 2 の危険な場所での使用を許可されていません。これらのスイッチは、クラス | ディビジョン 2、またはクラス | ゾーン 2 の場所で使用すると、スパークの危険または爆発の危険の可能性があります。

A.8 入力シミュレータ

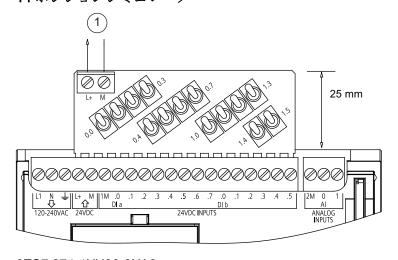
8ポジションシミュレータ



① DC 24 V センサ電源出力

6ES7 274-1XF30-0XA0

14 ポジションシミュレータ



① DC 24 V センサ 電源出力

6ES7 274-1XH30-0XA0

A.9 I/O 拡張ケーブル

技術データ	
注文番号(MLFB)	6ES7 290-6AA30-0XA0
ケーブル長	2 m
重量	200 g

I/O 拡張ケーブルにはオスコネクタとメスコネクタがあります。

- 1. オスコネクタをシグナルモジュールの右側にあるバスコネクタに接続します。
- 2. メスコネクタをシグナルモジュールの左側にあるバスコネクタに接続します。
 - メスコネクタのフック拡張をバスコネクタのハウジングに入れます。
 - メスコネクタをバスコネクタに押し込みます。

A.9 I/O 拡張ケーブル

電源容量の計算

CPU は、CPU 本体、任意の拡張モジュール、その他の DC 24 V ユーザー電力要件のための電源を供給する内部電源を備えています。

次の3つの種類の拡張モジュールがあります。

- シグナルモジュール(SM)は CPU の右側に取り付けます。 各 CPU には、電源容量 を考慮せず、取り付け可能な最大数のシグナルモジュールを取り付けることができます。
 - CPU 1214 には 8 個のシグナルモジュールを設置できます
 - CPU 1212 には 2 個のシグナルモジュールを設置できます
 - CPU 1211 にはシグナルモジュールを設置できません
- 通信モジュール(CM)は CPU の左側に取り付けます。 任意の CPU に、電源容量を 考慮せず、最大 3 個の通信モジュールを設置できます。
- シグナルボード(SB)は CPU の前面に取り付けます。 任意の CPU に、最大 1 個の シグナルボードを設置できます。

以下の情報を参考に、お客様のコンフィグレーションで CPU が提供できる電源(電力) を判断してください。

各 CPU は DC 5 V および DC 24 V の両方の電力を供給します。

- CPU は、1 個の拡張モジュールを接続すると、拡張モジュールに DC 5 V の電力を 供給します。 拡張モジュールの DC 5 V の電力要件が CPU の電源容量を超過する 場合、要件が電源容量内になるまで拡張モジュールを取り外す必要があります。
- 各 CPU には DC 24 V センサ電源があり、ローカル入力ポイントまたは拡張モジュールのリレーコイル用に DC 24 V を供給できます。 DC 24 V の電力要件が CPU の電源容量を超過する場合、外部 DC 24 V 電源を追加して、拡張モジュールに DC 24 V を供給することができます。 入力ポイントまたはリレーコイルに手作業で DC 24 V を接続する必要があります。

个警告

DC 24 V 外部電源を DC センサ電源と同時に接続すると、それぞれの電源が望ましい出力電圧レベルを確立しようとするので、2 つの電源間で衝突が発生する可能性があります。

この競合によって両方の電源の寿命が短くなったり、すぐに故障する原因となり、その結果、PLCシステムの予測できない動作が発生することがあります。 予測がつかない動作によって、死傷事故などの重大な人的傷害や物的損害が発生するおそれがあります。

CPU および外部電源の DC センサ電源によって、異なるポイントに電源を供給する必要があります。 共通の電源を 1 つ接続することができます。

PLC システムの一部の 24 V 電源入力ポートは、複数の M 端子を接続する共通の論理回路で相互接続されています。 CPU の 24 V 電源入力、SM リレーコイル電源入力、そして非絶縁のアナログ電源入力がデータシートで非絶縁として指定されている場合に相互接続されている回路の例です。 すべての非絶縁 M 端子は、同じ外部基準電圧に接続する必要があります。

! 警告

非絶縁 M 端子を異なる基準電位に接続すると、PLC および接続されている装置を損傷するか、またはこれらの予測できない動作の原因となる意図しない電流フローが発生します。

このような損傷または予測がつかない動作によって、死傷事故などの重大な人的傷害 や物的損害が発生するおそれがあります。

常に、PLCシステムのすべての非絶縁 M 端子を同じ基準電位に接続してください。

CPUの電源容量およびシグナルモジュールの電力要件についての詳細は、「技術仕様」 (ページ 391)に記載されています。

注記

CPU の電源容量を超過すると、CPU に許可されている最大モジュール数を接続できない場合があります。

B.1 電力要件の計算例

以下の例は、1 個の CPU 1214C AC/DC/リレー、3 個の SM 1223 8 DC 入力/8 リレー出力、1 個の SM 1221 8 DC 入力を含む PLC の電力要件の計算方法です。この例では、合計 46 の入力と 34 の出力があります。

注記

CPU は、内部リレーコイルを駆動するために必要な電力をすでに配分しました。 電源 容量の計算に、内部リレーコイルの電力要件を含める必要はありません。

この例で、CPU は SM に十分な DC 5 V 電流を供給しますが、すべての入力と拡張リレーコイルのセンサ電源からの十分な DC 24 V 電流を供給しません。 I/O には 448 mA 必要なのに、CPU は 400 mA しか供給しません。 この設置では、含まれるすべての DC 24 V 入力と出力を動作させるため、少なくとも DC 24 V 電源で 48 mA を供給する追加電源が必要です。

CPU の電源容量	DC 5 V	DC 24 V
CPU 1214C AC/DC/リレー	1600 mA	400 mA
	マイナス	
システム要件	DC 5 V	DC 24 V
CPU 1214C、14 入力	-	14 * 4 mA = 56 mA
SM 1223 × 3、5 V 電力	3 * 145 mA = 435 mA	-
SM 1221 × 1、5 V 電力	1 * 105 mA = 105 mA	-
SM 1223 × 3、各 8 入力	-	3 * 8 * 4 mA = 96 mA
SM 1223×3、各8リレー	-	3 * 8 * 11 mA = 264 mA
コイル		
SM 1221 × 1、8 入力	-	8 * 4 mA = 32 mA
統合要件	540 mA	448 mA
合計		
残りの電流	DC 5 V	DC 24 V
残りの合計電流	1060 mA	(48 mA)

B.2 電力要件の計算練習

B.2 電力要件の計算練習

以下の表を使用して、お客様のコンフィグレーションでS7-1200 のCPUが供給できる電力(電流量)を計算してみましょう。 ご使用のCPUモデルの電源容量とシグナルモジュールの電力要件については、「技術仕様」 (ページ 391)を参照してください。

CPU の電源容量	DC 5 V	DC 24 V	
	マイナス		
システム要件	DC 5 V	DC 24 V	
統合要件			
	合計		
残りの電流	DC 5 V	DC 24 V	
残りの合計電流			

注文番号

CPU		注文番号
CPU 1211C	CPU 1211C DC/DC/DC	6ES7 211-1AD30-0XB0
	CPU 1211C AC/DC/リレー	6ES7 211-1BD30-0XB0
	CPU 1211C DC/DC/リレー	6ES7 211-1HD30-0XB0
CPU 1212C	CPU 1212C DC/DC/DC	6ES7 212-1AD30-0XB0
	CPU 1212C AC/DC/リレー	6ES7 212-1BD30-0XB0
	CPU 1212C DC/DC/リレー	6ES7 212-1HD30-0XB0
CPU 1214C	CPU 1214C DC/DC/DC	6ES7 214-1AE30-0XB0
	CPU 1214C AC/DC/リレー	6ES7 214-1BE30-0XB0
	CPU 1214C DC/DC/リレー	6ES7 214-1HE30-0XB0

シグナルモジュー	ール、通信モジュール、シグナルボード	注文番号
シグナルモジュ	SM 1221 8×24V DC 入力	6ES7 221-1BF30-0XB0
ール	SM 1221 16×24V DC 入力	6ES7 221-1BH30-0XB0
	SM 1222 8×24V DC 出力	6ES7 222-1BF30-0XB0
	SM 1222 16×24V DC 出力	6ES7 222-1BH30-0XB0
	SM 1222 8 × リレー出力	6ES7 222-1HF30-0XB0
	SM 1222 16×リレー出力	6ES7 222-1HH30-0XB0
	SM 1223 8×24V DC 入力 / 8×24V DC 出力	6ES7 223-1BH30-0XB0
	SM 1223 16×24V DC 入力 / 16×24V DC 出力	6ES7 223-1BL30-0XB0
	SM 1223 8×24V DC 入力 / 8×リレー出力	6ES7 223-1PH30-0XB0
	SM 1223 16×24V DC 入力 / 16×リレー出力	6ES7 223-1PL30-0XB0
	SM 1231 4×アナログ入力	6ES7 231-4HD30-0XB0
	SM 1231 8×アナログ入力	6ES7 231-4HF30-0XB0
	SM 1232 2×アナログ入力	6ES7 232-4HB30-0XB0
	SM 1232 4×アナログ入力	6ES7 232-4HD30-0XB0
	SM 1234 4×アナログ入力 / 2×アナログ出力	6ES7 234-4HE30-0XB0
通信モジュール	CM 1241 RS232	6ES7 241-1AH30-0XB0
	CM 1241 RS485	6ES7 241-1CH30-0XB0
シグナルボード	SB 1223 2×24V DC 入力 / 2×24V DC 出力	6ES7 223-0BD30-0XB0
	SB 1232 1 アナログ出力	6ES7 232-4HA30-0XB0

HMI デバイス	注文番号
KTP400 Basic (Mono、PN)	6AV6 647-0AA11-3AX0
KTP600 Basic (Mono、PN)	6AV6 647-0AB11-3AX0
KTP600 Basic (Color、PN)	6AV6 647-0AD11-3AX0
KTP1000 Basic (Color、PN)	6AV6 647-0AF11-3AX0
TP1500 Basic (Color、PN)	6AV6 647-0AG11-3AX0

プログラムパッケージ	注文番号
STEP 7 Basic v10.5	6ES7 822-0AA0-0YA0

メモリカード、その他ハードウェア、スペアパーツ		注文番号
メモリカード	SIMATIC MC 2 MB	6ES7 954-8LB00-0AA0
	SIMATIC MC 24 MB	6ES7 954-8LF00-0AA0
その他ハードウェア	PSU 1200 電源	6EP1 332-1SH71
	CSM 1277 イーサネットスイッチ - 4 ポート	6GK7 277-1AA00-0AA0
	シミュレータ(1214C/1211C - 8 ポジション)	6ES7 274-1XF30-0XA0
	シミュレータ(1214C - 14 ポジション)	6ES7 274-1XH30-0XA0
	I/O 拡張ケーブル、2 m	6ES7 290-6AA30-0XA0
スペアパーツ	コネクタブロック、7 端子、スズ	6ES7 292-1AG30-0XA0
	コネクタブロック、8 端子、スズ(4 個/パッケージ)	6ES7 292-1AH30-0XA0
	コネクタブロック、11 端子、スズ(4 個/パッケージ)	6ES7 292-1AL30-0XA0
	コネクタブロック、12 端子、スズ(4 個/パッケージ)	6ES7 292-1AM30-0XA0
	コネクタブロック、14 端子、スズ(4 個/パッケージ)	6ES7 292-1AP30-0XA0
	コネクタブロック、20 端子、スズ(4 個/パッケージ)	6ES7 292-1AV30-0XA0
	コネクタブロック、3 端子、金(4 個/パッケージ)	6ES7 292-1BC0-0XA0
	コネクタブロック、6 端子、金(4 個/パッケージ)	6ES7 292-1BF30-0XA0
	コネクタブロック、7 端子、金(4 個/パッケージ)	6ES7 292-1BG30-0XA0
	コネクタブロック、11 端子、金(4 個/パッケージ)	6ES7 292-1BL30-0XA0

ドキュメント	注文番号
S7-1200 PLC システムマニュアル	
ドイツ語	• 6ES7 298-8FA30-8AH0
• 英語	• 6ES7 298-8FA30-8BH0
• フランス語	• 6ES7 298-8FA30-8CH0
• スペイン語	• 6ES7 298-8FA30-8DH0
• イタリア語	• 6ES7 298-8FA30-8EH0
• 中国語	• 6ES7 298-8FA30-8FH0
S7-1200 簡易マニュアル	
ドイツ語	• 6ES7 298-8FA30-8AQ0
• 英語	• 6ES7 298-8FA30-8BQ0
• フランス語	• 6ES7 298-8FA30-8CQ0
• スペイン語	• 6ES7 298-8FA30-8DQ0
• イタリア語	• 6ES7 298-8FA30-8EQ0
中国語	• 6ES7 298-8FA30-8FQ0

索引

Δ.	スタートアップパラメータ, 45, 83
A	スタートアップ処理, 48
AC	セキュリティレベル, 67
誘導負荷, 41	デバイスコンフィグレーション, 91
AND命令, 172	デバイスへのダウンロード, 309
ATEX承認, 393	トランスファーカード,83
ATTACH割り込み命令, 233	トランスファーカードを作成する, 83
	ネットワーク接続, 98
С	パスワードを忘れてしまった場合の復元, 68
6	パスワード保護,67
CAN_DINT時間遅延割り込み命令, 237	パラメータを設定する, 94
CE承認, 391	プログラムカード, 87
CM 1241 RS232 仕様, 442	プログラムカードを作成する, 87
CM 1241 RS485 仕様, 441	プログラムの実行, 44
CPU	モジュールを追加する, 95
1211C仕様, 398	モニタテーブル, 385
1211C配線図, 403	ランプ負荷, 42
1212C仕様, 404	動作モード, 46
1212C配線図, 410	取り付け手順,30
1214C仕様, 412	忘れてしまったパスワード, 68
1214C配線図, 417	接地工事, 39
HMIへの通信のコンフィグレーション, 310	新しいデバイスを追加する,92
IPアドレス, 99, 304	未指定のCPU, 93
MACアドレス, 327	概要, 11
PROFINET, 99, 304	比較チャート, 13
STOPモード, 388	空のトランスファーカード,68
イーサネットポート, 99, 304	絶縁のガイドライン, 39
オンライン, 382	複数のコンフィグレーション, 313
オンライン接続する, 381	誘導負荷, 41
オンライン操作パネル, 383	配線のガイドライン, 38, 40
サーマルゾーン, 28	電力要件, 447
サイクルタイム, 383	電源容量, 27

シグナルボード(SB), 15

C-Tick承認, 393 概要, 22 CTRL PWM命令, 243 HSC (高速カウンタ), 141 コンフィグレーション, 145 CTS, 335 CULUS認可, 392 I D I/O Date命令, 180 アドレス指定,75 DB (データブロック), 113 アナログステータスインジケータ,380 DC デジタルステータスインジケータ,380 誘導負荷,41 誘導負荷,41 I/O モジュール DEC (デクリメント)命令, 154 モニタテーブル,385 DETACH割り込み命令, 233 DIN レール, 29 INC (インクリメント)命令, 154 DIS AIRTアラーム割り込み命令, 240 IPアドレス, 99, 304 DTLデータタイプ, 80 オンラインでの割り付け,302 コンフィグレーションの実行,99,304 割り付け, 298, 307 Ε IPアドレス、オンラインCPUの設定, 382 EN_AIRTアラーム割り込み命令, 240 IPルータ, 100, 305 ENおよびENO (パワーフロー), 116 J F JMPN命令, 170 FB (ファンクションブロック), 111 FBD(ファンクションブロックダイアグラム), 116 L FC (ファンクション), 110 FM承認, 392 LAD (ラダーロジック), 115 Freeportプロトコル, 331 LEDインジケータ, 346, 379 Η M HMI MACアドレス, 327 PROFINET通信のコンフィグレーション, 310 MAX (最大値)命令, 156 HMIデバイス MB_COMM_LOAD, 264 ネットワーク接続,98 MB MASTER, 267

MB_SLAVE, 281 PtP通信, 331 **PWM** MIN (最小值)命令, 156 MOD (剰余算)命令, 152 CTRL PWM命令, 243 MODBUS, 264 MB MASTER, 267 R MB_SLAVE, 281 RCV_CFG (receive configuration)命令, 353 RCV_PTP (receive Point-to-Point)命令, 367 Ν RCV_RST (receiver reset)命令, 369 NEG (2の補数)命令, 153 RE TRIGR命令, 206 RS232 およびRS485 通信モジュール, 332 Network time protocol (NTP), 329 Not OK命令, 149 RT (タイマリセット)命令, 131 RTS, 335 RTSオフ遅延, 337 0 RTSオン遅延, 337 OK命令, 149 RTSが常にオン, 335 OR命令, 172 RTS切り替え, 335 RUNモード, 46, 49 RUNモードからSTOPモードへの移行,66 Р PID Compact命令, 241 S **PLC** システム設計, 103 S_CONV命令, 185 ブロックを使用する,105 S7-1200 概要,11 CMを取り付ける, 34 PLCシステムを設計する, 103, 105 CPU, 11 PORT_CFG (port configuration)命令, 348 CPUのパラメータを設定する,94 PROFINET, 99, 304 HMIデバイス, 22 IPアドレス, 99, 304 IPアドレス, 99, 304 ネットワークのテスト,307 PROFINET, 99, 304 ネットワーク接続,98 SBを取り付ける,36 PROFINET インターフェース SMを取り付ける, 32 イーサネットアドレスのプロパティ, 100, 305 イーサネットポート, 99, 304 時間同期のプロパティ, クリアランス.26 PTO (パルストレイン出力), 243 サーマルゾーン,28 PtPプログラミング, 344 サイクルタイム,

シグナルボード(SB), 15 シグナルモジュール(SM), 15 スタートアップパラメータ, 45, 83 デバイスコンフィグレーション, 91 トランスファーカード, 83 ネットワーク接続, 98 パスワード保護, 67 プログラムカード, 87

モジュールを設定する,96 モジュールを追加する.95

ランプ負荷,42

取り付けの概要,29

取り付け寸法,28

取り付け手順,30

各CPUモデルの比較, 13

忘れてしまったパスワード,68

接地工事,39

新しいデバイスを追加する,92

機能の拡張, 14

空のトランスファーカード,68

端子台コネクタ,37

絶縁のガイドライン,39

誘導負荷,41

通信モジュール(CM), 16

配線のガイドライン, 38, 40

電源容量,27

S7-1200機能の拡張, 14

SB 1223 仕様, 437, 439

SB 1223 配線図, 439

SEND_CFG (send configuration)命令, 351

SEND_PtP (send Point-to-Point data)命令, 363

SGN GET (get RS232 signals)命令, 369

SGN_SET (set RS232 signals)命令, 371

SM 1232 配線図, 440

SRT DINT時間遅延割り込み命令, 237

STEP 7

CPUのコンフィグレーション, 94

PROFINET, 99, 304

イーサネットポート, 99, 304

インストール, 16

デバイスコンフィグレーション,91

ネットワーク接続,98

プロジェクトビュー, 17

ポータルビュー, 17

モジュールを設定する,96

モジュールを追加する.95

新しいデバイスを追加する,92

STOPモード, 388

STP (PLCスキャンサイクルを停止する)命令, 207

STRG VAL命令, 185

Т

T_ADD命令, 180

T_CONV命令, 180

T_DIFF命令, 180

T_SUB命令, 180

TCON命令, 220

TCP/IP通信, 293

TDISCON命令, 220

TIA Portal

CPUのコンフィグレーション, 94

PROFINET, 99, 304

イーサネットポート, 99, 304

インストール, 16

デバイスコンフィグレーション,91

ネットワーク接続,98

プロジェクトビュー, 17

ポータルビュー, 17

モジュールを設定する,96

モジュールを追加する,95

新しいデバイスを追加する,92

アナログシグナルモジュール電圧,430 TOF (オフディレイ)タイマ命令, 131 TON (オンディレイ)タイマ命令, 131 TONR (保持型オンディレイ)タイマ命令, 131 イ TP (パルス遅延)タイマ命令, 131 イーサネット TRCV C命令, 213, 322 IPアドレス, 99, 304 TRCV_C命令のコンフィグレーション, 323 ネットワーク接続,98 TRCV命令, 220 イーサネット命令 TSAP(トランスポートサービスアクセスポイント), 318, TCON, 220 324 TDISCON, 220 TSEND C命令, 213, 316 TRCV, 220 TSEND_C命令のコンフィグレーション,317 TRCV_C, 213 TSEND命令, 220 **TSEND**, 220 TSEND C, 213 イーサネット通信, 293 U イベント実行,52 USS DRV命令, 252 インスタンスデータブロック,69 USS PORT命令, 256 インストール STEP 7, 16 USS_RPM命令, 257 TIA Portal, 16 USS_WPM命令, 260 USSステータスコード, 262 USSプロトコルライブラリ, 248 ウ ウォッチドッグ, 206 V 工 VAL_STRG命令, 185 エラー PtP命令, 372 X タイムエラー,55 XON / XOFF, 336 診断エラー,56 エンコード(ENCO)命令, 173 XOR (排他的OR)命令, 172 ア 才 オーガニゼーションブロック アイドルライン, 338, 339 CALL, 50 アナログI/Oステータスインジケータ,380 ファンクション,50 アナログシグナルモジュールの仕様,427

作成,110 カウンタ命令, 135 優先度クラス,50 カスタマサポート,4 処理, 108 カレンダ命令, 180 動作を設定する,110 複数の周期. 110 キ オフディレイ(TOF)命令, 131 キュー, 52 オンディレイ(TON)命令, 131 オンライン、オンライン接続する,381 オンラインCPU, 382 サイクルタイムのモニタ,383 グローバルデータブロック, 69, 113 メモリ使用率のモニタ,383 グローバルライブラリ USS, 248 操作パネル, 383 オンラインヘルプ, 18 ヘルプウィンドウを拡大する,20 ク 印刷. 21 クロック 独立表示. 20 時計,63 目次と索引を表示する,20 クロック命令, 183 オンラインヘルプウィンドウを拡大する,20 システム時刻を書き込む(WR_SYS_T), 183 オンラインヘルプヘアクセス, 18 システム時刻を読み取る(RD SYS T), 183 オンラインヘルプを独立して表示する,20 ローカル時刻を読み取る(RD_LOC_T), 183 ガ コ ガイドライン コネクタ ランプ負荷,42 取り付けと取り外し,37 取り付け手順,29 コピー保護, 117 接地工事,39 コンフィグレーション 絶縁,39 HMIからCPU, 312 設置, 25 HSC (高速カウンタ), 145 誘導負荷,41 IPアドレス, 99, 304 配線のガイドライン,38,40 PLCからPLCへの通信, 313 PROFINET, 99, 304 サイクルタイム, 力 スタートアップパラメータ, 45, 83 カウンタ ポート. 333 高速(HSC), 141 受信メッセージ,338 高速(HSC): コンフィグレーションの実行, 145

ジ 検出する,93 産業用イーサネットポート,99,304 ジャンプ(JMP)命令, 170 通信インターフェース,333 シ サ シリアル通信,331 サーマルゾーン,28 シングルインスタンス 例, 112 サイクルタイム,59 サイクルタイムのモニタ、オンラインCPU, 383 サブネットマスク, 100, 305 ス サポート.4 スキャンサイクルタイム.58 スケーリング(SCALE X)命令, 168 シ スタートアップパラメータ, 45, 83 シグナルボード(SB) ストップビット,334 スレーブポーリングアーキテクチャ,345 デバイスコンフィグレーション,91 モジュールを追加する.95 スワップ命令, 164 取り外し,36 新しいデバイスを追加する,92 セ 概要, 15 セキュリティ 比較チャート, 14 **CPU, 67** 設置.36 プログラムブロック, 117 電力要件. 447 忘れてしまったパスワード,68 シグナルモジュール セット命令, 125 SM 1221 仕様, 419 SM 1222 仕様, 420 そ SM 1223 仕様, 423 シグナルモジュール(SM) その他のPtPパラメータエラー, 377 デバイスコンフィグレーション,91 モジュールを追加する,95 ソ 取り外し,32 ソフトウェアフロー制御,336 新しいデバイスを追加する,92 概要, 15 比較チャート, 14 タ 設置,32 タイマリセット(RT)命令, 131 電力要件, 447 タイマ命令, 131

ツ 卜 ツールチップ, 18 トランスファーカード,83 スタートアップパラメータを設定する,83 忘れてしまったパスワード,68 デ 空のトランスファーカード,68 データタイプ,80 DTL, 80 ネ STRING, 78 配列. 79 ネガティブエッジ命令, 128 データブロック ネットワーク接続,98 インスタンスデータブロック.69 複数のCPU, 315 オーガニゼーションブロック(OB), 108 ネットワーク接続を作成する,98 グローバルデータブロック, 69, 113 ネットワーク通信, 295 データブロック (DB), 113 データ処理ブロック(DHB), 113 データ転送、開始. 363 ノウハウ保護, 117 デコード(DECO)命令, 173 デジタルI/Oステータスインジケータ,380 デジタルシグナルボード(SB)の仕様, 437 ハ デバイスコンフィグレーション,91 ハードウェアコンフィグレーション,91 CPUのコンフィグレーション,94 CPUのコンフィグレーション,94 PROFINET, 99, 304 PROFINET, 99, 304 イーサネットポート, 99, 304 イーサネットポート, 99, 304 ネットワーク接続,98 ネットワーク接続,98 モジュールを設定する,96 モジュールを設定する,96 モジュールを追加する,95 モジュールを追加する,95 新しいデバイスを追加する,92 新しいデバイスを追加する,92 検出する,93 検出する,93 デバイスへのダウンロード,309 ハードウェアフロー制御,335 デバイスを挿入する 未指定のCPU, 93 デバイスを追加する は 未指定のCPU, 93 はじめに オンラインヘルプ, 18 ド ツールチップ. 18 ツールチップを表示する, 18 ドキュメント, 18

ドキュメント, 18 ファンクションブロック(FB) ポータルビューとプロジェクトビュー, 17 インスタンスデータブロック,111 ロールアウトヘルプ, 18 出力パラメータ, 111 情報システム, 18 初期值, 111 状況に応じたヘルプ. 18 フィル(FILL BLK)命令, 162 バ ブ ブレーク信号,338 バスコネクタ, 15 パ フ パスワード,68 フロー制御,336 パスワード保護 コンフィグレーション, 334 CPU, 67 プログラムブロック, 117 プ 忘れてしまったパスワード,68 プログラミング 空のトランスファーカード.68 FBD(ファンクションブロックダイアグラム), 116 パラメータのコンフィグレーション LAD (ラダー), 115 受信, 323 PtP命令, 344 送信, 317 パラメータを設定する パワーフロー(ENおよびENO), 116 CPU, 94 リニア, 106 PROFINET, 99, 304 未指定のCPU, 93 イーサネットポート, 99, 304 構造化, 106 モジュール.96 プログラムカード,87 パラメータ割り付け,111 スタートアップパラメータを設定する,83 パリティ**, 334** プログラムの実行,44 パルストレイン出力(PTO), 243 プログラムブロック, 117 パルス遅延(TP)命令, 131 DB (データブロック), 113 FB (ファンクションブロック), 111 FC (ファンクション), 110 F. ノウハウ保護, 117 ビットロジック,121 プログラムをモニタする, 120 プログラムを試験する, 120 フ プログラム構造, 107 プロジェクト ファンクション(FC), 110

CPUへのアクセスを制限する, 67 トランスファーカード,83 ヘルプ, 18 プログラムカード,87 印刷. 21 プログラムブロックを保護する,117 拡大, 20 忘れてしまったパスワード,68 独立表示, 20 空のトランスファーカード,68 目次と索引を表示する,20 プロジェクトビュー, 17 ヘルプトピックの印刷, 21 CPUのパラメータを設定する,94 PROFINET, 99, 304 ポ イーサネットポートを設定する,99,304 ポイントツーポイントプログラミング,344 デバイスコンフィグレーション,91 ポイントツーポイント命令の戻り値,372 ネットワーク接続,98 ポイントツーポイント通信,331 モジュールを設定する,96 ポータルビュー, 17 モジュールを追加する.95 CPUのコンフィグレーション,94 新しいデバイスを追加する.92 PROFINET, 99, 304 イーサネットポートを設定する,99,304 ブ モジュールを設定する,96 ブロック モジュールを追加する,95 オーガニゼーションブロック(OB), 108 新しいデバイスを追加する,92 タイプ, 43 ポートコンフィグレーション,333 データブロック(DB), 43 命令, 344 ファンクション(FC), 43 ポートコンフィグレーションエラー, 373 ファンクションブロック(FB), 43 ポーリングアーキテクチャ.345 ブロックのムーブ(MOVE_BLK)命令, 160 ブロック呼び出し シングルインスタンスまたはマルチインスタンスと ボ して呼び出す, 111 ボーレート,334 基本, 43 ポ プ ポジティブエッジ命令, 128 プロトコル freeport, 331 通信, 331 ホ ホットライン,4

メモリ使用率のモニタ、オンラインCPU, 383 マスタポーリングアーキテクチャ,345 モ モジュール A サーマルゾーン,28 ムーブ命令, 160 シグナルボード(SB), 15 シグナルモジュール(SM), 15 メ パラメータを設定する,96 メッセージのコンフィグレーション 比較チャート, 14 受信, 338 通信モジュール(CM), 16 命令, 344 モニタテーブル,385 送信, 337 メッセージ終了,341 ラ メッセージ終了文字, 341 ラベル命令, 170 メッセージ長, 341 ランプ負荷,42 メッセージ開始, 339 メッセージ開始文字, 339 メモリ IJ I(プロセスイメージ入力), 71 リセット命令, 125 L(ローカルメモリ), 69 リニアプログラミング, 106 M (ビットメモリ), 73 リレーの電気耐用年数,397 Q(プロセスイメージ出力),72 クロックメモリ,64 ル システムメモリ,64 テンポラリメモリ,74 ルータのIPアドレス, 100, 305 ロードメモリ,60 ワークメモリ.60 口 保持メモリ,60 メモリカード ロールアウトヘルプ, 18 スタートアップパラメータを設定する,83 トランスファーカード,83 プログラムカード,87 一般技術仕様), 391 忘れてしまったパスワード,68 空のトランスファーカード,68 メモリカードの仕様,443

メモリロケーション, 69, 71

上 入力シミュレータ,443 定格電圧, 397 上下限リミット命令, 157 海事承認, 393 環境, 394 丸 環境条件, 395 丸め命令, 166 通信モジュールCM 1241 RS232, 442 通信モジュールCM 1241 RS485, 441 配線図: SM 1231、1232、1234, 435 乗 電磁環境両立性(EMC), 394 乗算(MUL)命令, 150 保 仕 保持型オンディレイ(TONR)命令, 131 仕様 保護クラス,396 ATEX承認, 393 保護レベル CE承認, 391 CPU, 67 CPU 1211C, 398 プログラムブロック, 117 CPU 1212C, 404 忘れてしまったパスワード,68 CPU 1214C, 412 C-Tick承認, 393 信 CULUS認可, 392 FM承認, 392 信号処理エラー, 375 SB 1223, 439 SB 1223, 439 値 SM 1221 シグナルモジュール, 419 値から文字列への変換命令, 185 SM 1221 配線図, 420 SM 1222 シグナルモジュール, 420 SM 1222 配線図, 422 優 SM 1223 シグナルモジュール, 423 優先度クラス SM 1223 配線図, 426 概要,50 アナログシグナルモジュール,427 アナログシグナルモジュール電圧,430 入 デジタルシグナルボード(SB), 437 入力シミュレータ,443 メモリカード,443 リレーの電気耐用年数,397 一般技術, 391

保護,396

処

処理の優先度,52

出

出力パラメータ, 111

切

切り上げ(CEIL)命令, 167 切り下げ(FLOOR)命令, 167 切捨て(TRUNC)命令, 166

割

割り込み

概要,50

割り込み不可能なフィル(UFILL_BLK)命令, 162 割り込み不可能なムーブ(UMOVE_BLK)命令, 160 割り込み待ち時間. 55

加

加算(ADD)命令, 150

反

反転(INV)命令, 172

取

取り付け

CPU, 30

ガイドライン,25

クリアランス, 26

サーマルゾーン,28

シグナルボード(SB), 36

シグナルモジュール(SM), 32

ランプ負荷,42

寸法, 28

接地工事,39

概要, 29

端子台コネクタ,37

絶縁,39

誘導負荷,41

通信モジュール(CM), 34

配線のガイドライン, 38, 40

取り付け時に確保する空間.26

受

受信コンフィグレーションエラー, 374 受信パラメータのコンフィグレーション, 323 受信メッセージのコンフィグレーション, 338 受信ランタイムの戻り値, 376

右

右シフト(SHR)命令, 177 右ローテーション(ROR)命令, 178

各

各モジュールの比較チャート, 14

命

命令

AND, 172

CTRL_PWM), 243

DEC (デクリメント), 154

GET ERROR, 213

INC (インクリメント), 154

MAX (最大値), 156

MIN (最小值), 156

MOD (剰余算), 152 カウンタ,135 カレンダ, 180 NEG (2 の補数), 153 クロック, 183 OK, 149 OKでない, 149 クロック:システム時刻を書き込む OR, 172 (WR_SYS_T), 183 PID_Compact, 241 クロック:システム時刻を読み取る PORT_CFG (port configuration), 348 (RD_SYS_T), 183 RCV_CFG (receive configuration), 353 クロック: ローカル時刻を読み取る RCV_PtP (receive Point-to-Point), 367 (RD_LOC_T), 183 RCV RST (receiver reset), 369 ジャンプ(JMP), 170 RE_TRIGR, 58, 206 スケーリング(SCALE_X), 168 RESET, 125 スワップ, 164 ROUND, 166 タイマ, 131 SEND_CFG (send configuration), 351 タイマ: RT (リセットタイマ), 131 SEND PTP (send Point-to-Point data), 363 タイマ: TOF (オフディレイタイマ), 131 SET, 125 タイマ: TON (オンディレイタイマ), 131 タイマ: TONR (保持型オンディレイタイマ), 131 SGN_GET (get RS232 signals), 369 タイマ: TP (パルスタイマ), 131 SGN_SET (set RS232 signals), 371 STP (PLCスキャンサイクルを停止する), 207 デコード(DECO), 173 ネガティブエッジ, 128 T_ADD, 180 ビットロジック, 121 T_CONV, 180 T DIFF, 180 フィル(FILL BLK), 162 ブロックのムーブ(MOVE_BLK), 160 T_SUB, 180 TCON, 220 ポジティブエッジ, 128 ムーブ, 160 TDISCON, 220 ラベル, 170 TRCV, 220 TRCV C, 213 乗算(MUL), 150 **TSEND**, 220 値を文字列に S_CONV, 185

TSEND_C, 213 値を文字列に VAL_STRG, 185 USS_DRV, 252 切り上げ, 167 USS_PORT, 256 切り下げ(FLOOR), 167

USS_WPM, 260

XOR (排他的OR), 172 エンコード(ENCO), 173

USSステータスコード, 262

USS RPM, 257

切捨て(TRUNC), 166 割り込み: ATTACH, 233 割り込み: CAN_DINT, 237 割り込み: DETACH, 233 割り込み: SRT_DINT, 237 割り込み: DIS_AIRT, 240 割り込み: EN_AIRT, 240 割り込み不可能なフィル(UFILL_BLK), 162 割り込み不可能なムーブ(UMOVE_BLK)命令, 160 加算(ADD), 150 反転(INV), 172 右シフト(SHR), 177 右ローテーション(ROR), 178 変換, 165 多重化(MUX), 175 左シフト(SHL), 177 左ローテーション(ROL), 178 戻り値(RET), 171 文字列を値に S_CONV, 185 文字列を値に STRG_VAL, 185 日付, 180

時刻, 180 正規化(NORM), 168 比較命令, 147

浮動小数点演算, 158

減算(SUB), 150 範囲内, 148

範囲外, 148

絶対値(ABS), 155

選択(SEL), 175

限界值, 157

除算(DIV), 150

高速カウンタ(HSC), 138

変

変換命令, 185

多

多重化(MUX)命令, 175

定

定格電圧, 397

左

左シフト(SHL)命令, 177 左ローテーション(ROL), 178

待

待機時間, 334

忘

忘れてしまったパスワード,68

情

情報システム, 18

印刷, 21

拡大, 20

独立表示, 20

目次と索引を表示する,20

戻

戻り値

PtP命令, 372

戻り値(RET)命令, 171

技

技術サポート,4 技術仕様,391

担

担当窓口,4

改

改行, 339

数

数値

REAL, 77 浮動小数点, 77

文

文字シーケンス メッセージ終了,342 メッセージ開始,340 文字の位置 メッセージ長,342 文字列から値への変換命令,185 文字列データタイプ,78 文字間ギャップ,341

日

日付と時刻データタイプ,80

時

時刻、オンラインCPUの設定, 382 時刻命令, 180

最

最大メッセージ長,341

未

未指定のCPU, 93

検

検出する,93

構

構造化プログラミング, 106, 107

正

正規化(NORM)命令, 168

比

比較チャート CPUモデル, 13 HMIデバイス, 22 比較命令, 147

浮

浮動小数点演算命令, 158

海

海事承認, 393

減

減算(SUB)命令, 150

状

状況に応じたヘルプ, 18

環

環境

産業, 394 環境条件, 395

目

目次と索引を表示する(オンラインヘルプ), 20

端

端子台コネクタ 設置, **37**

算

算術演算命令, 150

範

範囲内, 148 範囲外命令, 148

終

終了条件, 341

絶

絶対値(ABS)命令, 155 絶縁のガイドライン, 39

設

設置

CPU, 30

ガイドライン, 25

クリアランス, 26

サーマルゾーン, 28

シグナルボード(SB), 36

シグナルモジュール(SM), 32

ランプ負荷,42

取り付け寸法,28

接地工事,39

概要, 25

端子台コネクタ,37

絶縁のガイドライン,39

誘導負荷,41

通信モジュール(CM), 34

配線のガイドライン, 38, 40

電源容量,27

診

診断バッファ,62,384

誘

誘導負荷,41

送

送信コンフィグレーションエラー, 374 送信パラメータのコンフィグレーション, 317 送信ブロック(Tブロック), 315 送信メッセージのコンフィグレーション, 337 送信ランタイムエラー, 375

通

通信

IPアドレス, 99, 304 ネットワーク, 295 ハードウェア接続, 296

フロー制御, 335

ポーリングアーキテクチャ,345

ライブラリ,331

ロード.59

送信パラメータと受信パラメータ.337

通信インターフェース

コンフィグレーション,333

プログラミング,344

通信モジュール RS232 およびRS485, 332 デバイスコンフィグレーション, モジュールを追加する,95 新しいデバイスを追加する,92 通信モジュール(CM), 16 データ受信,367 仕様,441 取り外し,34 概要, 16 比較チャート,14 設置,34 電力要件, 447 通信モジュール(CM)、USSライブラリ, 249 選 選択(SEL)命令, 175 配 配線のガイドライン 前提条件,38 接地工事,39 配線図

メッセージ, 342 長さm, 342 長さn, 342 **開** 開始条件, 339

除

除算(DIV)命令, 150

電

電力要件 計算, 449, 450 電源容量, 27 例, 449, 450 電磁環境両立性(EMC), 394

高

高速カウンタ, **141** 高速カウンタ(HSC)命令, **138**

長

長さ

CPU 1211C, 403 CPU 1212C, 410 CPU 1214C, 417

SM 1221 シグナルモジュール, 420 SM 1222 シグナルモジュール, 422 SM 1223 シグナルモジュール, 426

SM 1231、1232、1234, 435

SB 1223, SB 1232,