SIEMENS

SIMATIC

STEP 7
S5からS7への変換

User Manual

Reference Manual

Programming Manual

コンバータマニュアル

C79000-G7076-C551-01

安全の手引き

このマニュアルには、ユーザの安全を守るため、また、製品や接続された機器を保護するために守らなければならない注意事項が記載されています。注意事項は三角形の警告マークで強調されており、危険度によって次のような等級に分類されています。

危険

適切な注意が払われない場合、極めて高い可能性で、人に致命傷あるいは重傷を及ぼしたり、機器に重大な損傷を与える恐れがあります。

警告

適切な注意が払われない場合、人に致命傷あるいは重傷を及ぼしたり、機器に重大な損傷を与える恐れがあります。

注意

適切な注意が払われない場合、人に傷害を及ぼしたり、危機に損傷を与える恐れがあります。

注

製品とその取り扱い方法や、マニュアルの該当部分に関する重要な情報を記載しています。

資格のあるスタッフ

機器/システムのセットアップと操作は、必ずこのマニュアルに従って行わなくてはいけません。機器の取り付けと作業は、必ず資格のあるスタッフが行ってください。**資格のあるスタッフ**とは、安全基準に従って機器とシステムの配線と接地を行う資格のあるスタッフです。

正しい用途

以下の点に注意して下さい。

警告

この製品とそのコンポーネントは、カタログまたは説明書に記載されている用途にのみ使用可能であり、またシーメンスが許可あるいは推薦するメーカーの機器やコンポーネントとの接続においてのみ使用可能です。

この製品は、輸送、保管、セットアップ、取り付けが正しく行われ、適切な操作とメンテナンスが行われた場合にのみ、安全かつ正確に機能します。

商標

SIMATIC と SINEC は、SIEMENS AGの登録商標です。

第三者が、このマニュアルに記載されている商標に属する名称を許可なく使用した場合、商標権を侵害する可能性があります。

Siemens AG
Bereich Automatisierungstechnik
Industrial Automation Systems
Postfach 4848, D-90327 Nuernberg

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Siemens Aktiengesellschaft **C79000-G7076-C551**

まえがき

目的

このマニュアルは、S5 プログラムを S7 プログラムに変換する方法を説明しています。この説明に従えば、以下を実行できます。

- 既存の S5 プログラムを S7 プログラムに変換し、必要に応じてこれらを手動で編集する
- 変換済みの S7 ファンクション（以前の S5 標準ファンクションブロック）を S7 プログラムに組み込む

対象読者

既存の S5 プログラムを S7 で使用するプログラマを対象としています。

適用範囲

このマニュアルは、STEP 7 プログラミングソフトウェアのバージョン4.0に適用されます。

S7 関連資料の概要

S7 プログラマブルコントローラのコンフィグレーションやプログラミングに関しては、ユーザをサポートする資料が広範囲にわたって提供されています。以下の説明を参照して、必要な資料を判断することができます。

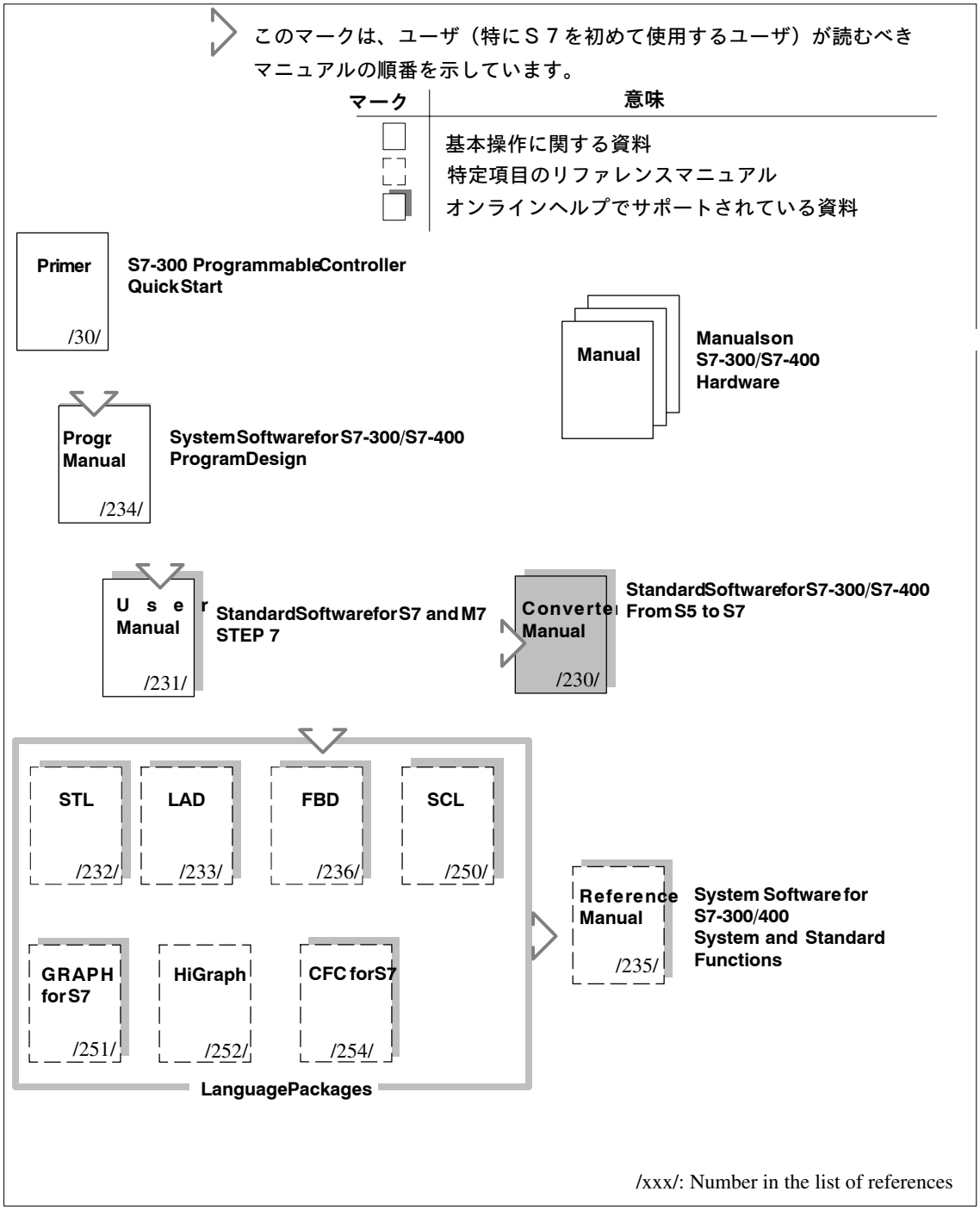

図 1-1　　S7 関連資料の概要

表 1-1　　S7 関連資料の概要

マニュアル名	内容
S7-300 Programmable Controller Quick Start, Primer	S7-300/400の構造とプログラミングの基礎が紹介されています。これは、S7プログラマブルコントロールシステムを初めて使用するユーザに適しています。
S7-300 and S7-400 Program Design Programming Manual	S7CPUのユーザプログラムとオペレーティングシステムの構造に関する基本的な情報が提供されています。S7-300/400 を初めて使用するユーザは、このマニュアルを使ってプログラミングの概要を把握し、ユーザプログラム作成の基礎を理解してください。
S7-300 and S7-400 System and Standard Functions Reference Manual	S7CPUには、システムファンクションとオーガニゼーションブロックのほかにそのオペレーティングシステムが組み込まれ、プログラミングの際にこれらを使用することができます。このマニュアルでは、S7で使用できるシステムファンクション、オーガニゼーションブロック、ロード可能な標準ファンクションの概要が示されています。また、ユーザプログラムでこれらのファンクションやブロックをどのように利用するかについても、リファレンス形式で詳細に説明されてます。
STEP 7 User Manual	STEP 7オートメーションソフトウェアの主な使い方と機能が説明されています。このマニュアルは、STEP7を初めて使用するユーザやSTEP5で経験を積んだユーザに、S7-300/400 でのコンフィグレーション、プログラミング、スタートアップの手順の概要を説明しています。 STEP7で作業を行う場合は、オンラインヘルプでもSTEP7の使用をサポートする情報が提供されています。
Converter Manual From S5 to S7	既存の STEP 5プログラムを変換して S7 CPU で実行したい場合は、このマニュアルが必要です。このマニュアルでは、コンバータの使用方法に関する概要を説明しています。コンバータの機能に関する詳細は、オンラインヘルプでも提供されています。また、オンラインヘルプでは、変換された S7 ファンクションのインターフェイスについても説明しています。さらに、SIMATICS7ハードウェアとソフトウェアの実際的な情報も提供しています。
Statement List, Ladder Logic, Function Block Diagram, SCL[1] Manuals	プログラミング言語であるステートメントリスト（STL）、ラダーロジック（LAD）、シーケンス制御言語（SCL）のマニュアルでは、概要説明とプログラミング言語の詳細が提供されています。S7-300/400 のプログラミングでは、1つの言語で十分ですが、必要であれば同一のプロジェクト内で複数の言語を使用することも可能です。これらの言語を初めて使用する場合には、このマニュアルに従って、最初に選択した言語での基本的なプログラム作成方法を習得することをお薦めします。 STEP7での作業中は、オンラインヘルプを使ってエディタやコンパイラの使い方に関する詳細を得ることができます。
GRAPH[1] , HiGraph[1], CFC[1] Manuals	GRAPH、HiGraph、CFC などの言語により、シーケンス制御、ステートグラフ、チャートの形式でブロックをプログラミングすることができます。このマニュアルでは、概要説明と、プログラミング言語の詳細が提供されています。これらの言語を初めて使用する場合は、最初に選択した言語での基本的なプログラム作成方法を習得することをお薦めします。 STEP7での作業中は、オンラインヘルプを使ってエディタやコンパイラの使い方に関する詳細を得ることができます（ただし HiGraph は除く）。

1　S7-300/S7-400 のシステムソフトウェアのオプショナルパッケージ

このマニュアルの構成

このマニュアルを使用するには、S7 プログラムに関する知識が必要です。これについては、「*Programming Manual* **/234/**」で説明されています。さらに、「*User Manual* **/231/**」に記載されている標準ソフトウェアを習得している必要があります。

このマニュアルは以下のように構成されています。

- 第 1 部（第 1 ~ 3 章）では、S5 から S7 への変換を計画する方法を説明します。
- 第 2 部（第 4 ~ 9 章）では、コンバータを使ってプログラムを変換する方法を説明します。
- 第 9 章では応用例を説明します。
- 付録はリファレンスセクションで、すべての STL 命令（国際表記と SIMATIC 表記）をリストします。
- 用語解説では重要な用語を説明します。
- 索引により、重要な項目を素早く簡単に見つけることができます。

関連マニュアル

その他の関連マニュアルは、マニュアルの部品番号をスラッシュ /.../ で囲んで示してあります。この番号により、本マニュアルの巻末にあるリファレンスセクションでマニュアル名を確認することができます。

その他のサポート

本書のソフトウェアの説明について、本書やオンラインヘルプでは解決できない疑問点がありましたら、最寄りの Siemens の担当部署までご連絡ください。連絡先につきましては、**/70/** や **/100/** の付録、カタログ、あるいは Compuserve（**`go autforum`**）で調べることができます。また、下記のホットラインでもご質問を受け付けています

Tel. (+49) (911) 895 7000 (Fax 7001)

このマニュアルについてご質問またはご意見がある場合は、巻末のコメント用紙にご記入になり、用紙に記載の住所までお送りください。その際、アンケートにもお答えいただければ幸いです。

弊社では、SIMATIC S7 オートメーションシステムを初めてご使用になる方のためのトレーニングコースを多数開催しています。詳細につきましては、最寄りのトレーニングセンターまたは中央トレーニングセンターまでお問い合わせください。

D-90327 Nuremberg, Tel. (+49) (911) 895-3154.

注記

このマニュアルは旧マニュアル名「Converting S5 Programs」から名称変更されたものです。他のマニュアルでは、この旧マニュアル名が使用されていたり、あるいは単に「Converter Manual」と記載されている場合があります。

目次

Teil 1: Planung des Umstiegs 第1部: 変換の計画

はじめに

1

これまでは、SIMATIC は SIEMENS プログラマブルコントローラ S5 ファミリの名称としてご存知だったでしょう。しかしここでは、SIMATIC は「完全に統合されたオートメーション」という意味で使用されています。

完全に統合されたオートメーションというコンセプトは、製造とプロセスエンジニアリングの世界をまったく新しい方法で結合したものです。すべてのハードウェアコンポーネントとソフトウェアコンポーネントが1つのシステム（SIMATIC）に統合されています。

このように完全な統合は、S7 システムが提供する広範な互換性によって可能になっています。この互換性は次の3つの分野で実現されています。

- データベース

  データを1度入力すると、その後は工場全体で使用できるようになります。したがって、転送エラーや不整合は過去のものとなりました。

- コンフィグレーションとプログラミング

  タスク内のコンポーネントとシステムの計画、コンフィグレーション、プログラミング、稼動、デバッグ、モニタリングは、完全に統合されたソフトウェアパッケージで行えます。このソフトウェアパッケージはモジュール形式で、ユーザインターフェース1つと、最適なユーティリティを備えています。

- 通信

  「通信元と通信相手」は接続テーブルで簡単に指定でき、いつでも変更することができます。各種のネットワークタイプを一定の方法で簡単にコンフィグレーションできます。

SIMATIC が完全に統合されたシステムとして様々な可能性を満たすことができるよう、SIMATIC S7 には最新のコンセプトが実現されています。このため、一部のファンクションは、S5 とは異なる方式で実行されます。

STEP 7 プログラミングソフトウェアは、新しいテクノロジとコンセプトを基本としています。たとえば、ユーザインターフェースは、人間工学的な要求事項を満たすように設計され、Windows 95/NT で実行します。プログラミング言語では、IEC 1131 規格にできる限り準拠し、STEP 5との互換性も保つように努めました。

新しい STEP 7 システムは次の要求事項を満たしています。

- 完全に統合されたオートメーションに対応するソフトウェア基盤
- IEC 1131 に準拠するプログラミング
- STEP 5 との互換性

既存のシステムから新しいシステムへの変換に際しては多数の疑問点が生じます。また、特にソフトウェアなどでは適合作業が必要になります。

このマニュアルは、これらの疑問点に答える同時に、STEP 5 プログラムを SIMATIC S7 で引き続き使用できるようにする簡単な方法を説明します。

ハードウェア

2

この章では、S5 から S7 へ容易に移行できるように、S7 で使用できるハードウェアを説明し、必要に応じて S5 で使用するハードウェアとの比較を行います。

CD-ROMのSiemensカタログによりハードウェアをS5からS7に変換

Siemens CD-ROM “Components for Automation” / catalog CA01（from 4/97）には、S5 から S7 への変換時にハードウェアを選択するためのアプリケーションが収録されています。製品カタログにアクセスするには、メニューコマンド [**Auswahlhilfen > Simatic**] を選択します。ここで、必要な S5 システムを入力することができます。このシステムデータを基に、アプリケーションでラックコンフィグレーションとシグナルリストが作成されます。その後、この S5 コンフィグレーションを S7 コンフィグレーションに変換できます。

2.1 プログラマブルロジックコントローラ

SIMATIC S7 プログラマブルロジックコントローラは、パフォーマンスレンジ別に次の3タイプに分類されます。

SIMATIC S7-200

SIMATIC S7-200は、小型のマイクロプログラマブルロジックコントローラ（PLC）で、3タイプの中では最もパフォーマンスレンジが低い用途に適しています。S7-200 は、システム固有の独自のソフトウェアパッケージで制御されます。このソフトウェアパッケージは、後述する S5 と S7 の比較では省略されています。

SIMATIC S7-300

SIMATIC S7-300 はモジュール形式のミニコントローラで、それほどパフォーマンスレンジの高くない用途に適しています。

SIMATIC S7-400

SIMATIC S7-400 は、パフォーマンスレンジが中レベルから高レベルの用途に適しています。

わかりやすいように、S7-300 モジュール名は “3” で開始し、S7-400 モジュール名は “4” で開始します。

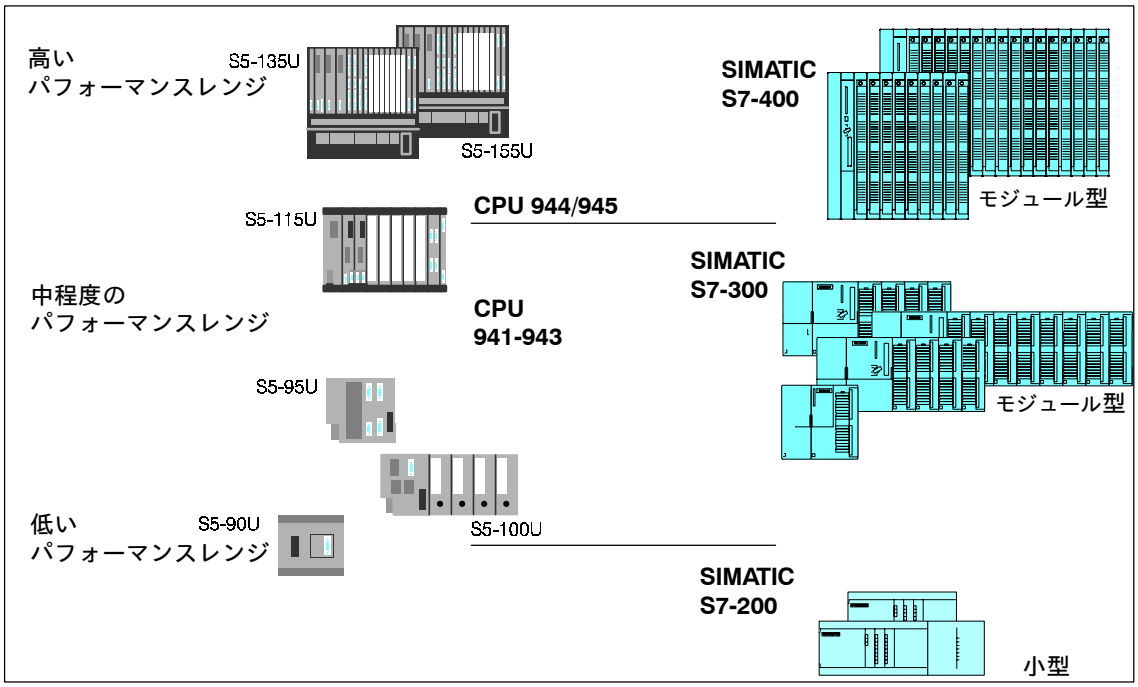

図 2-1 SIMATIC プログラマブルコントローラ

プログラミング装置とOPをSIMATIC S7に接続する

プログラミング装置およびオペレータパネル用のプログラミング装置インターフェース MPI（マルチポイントインターフェース）

SIMATIC S5 で使用されていたプログラミング装置インターフェースAS511は、マルチポイントインターフェース MPI（S7-300とS7-400用）に代わりました。このマルチポイントインターフェースにより、HMIHuman Machine Interface、旧名称は COROS）装置およびプログラミング装置を、SIMATIC S7 のプログラミング装置インターフェースに直接接続することができます。このインターフェースは完全に統合されています。

この2つのインターフェースの仕様を次の表で比較します。

AS511	MPI
25ピン TTY インターフェース（20mA）	9ピンDサブ、RS485インターフェース
転送レート: 9.6Kbps	転送レート: 187.5 Kbps
プロトコル: 3964R	プロトコル: S7 ファンクション
	ネットワークの最大距離: 50m（バス増幅器または特殊ケーブルを使用した場合は 1000m まで） システム内のプログラマブルモジュールは、マルチポイントインターフェースを介してアドレス指定することができます。
接続可能な装置は1個	装置は 31 個まで接続可能

オペレータパネル（OP）のバスインターフェース

SIMATIC S5 および SIMATIC S7 オートメーションファミリのプログラマブルロジックコントローラは、**PROFIBUS**（旧名称は SINEC L2 ）バスシステムを使って接続できます。以前と同様、バス固有の接続になります。

2.2 S7 モジュール

S5 と基本的に同等

S7 で使用するモジュール群は、SIMATIC S5 の従来のモジュールコンセプトに対応し、それをさらに拡張しています。

S7 には次のタイプのモジュールがあります。

- 中央演算装置（CPU）
- 電源モジュール（PS）
- インターフェースモジュール（IM）
- コミュニケーションプロセッサ（CP）;（PROFIBUS との接続など）
- ファンクションモジュール（FM）;（カウント、位置決め、クローズドループ制御など）
- デジタル/アナログモジュールは「シグナルモジュール（SM）」に名称変更

この章では、SIMATIC S5 と SIMATIC S7 で使用されるモジュールの類似点と相違点を説明します。

新しいパフォーマンス特性

S7 モジュール独自の機能として次のようなものがあります。

- ジャンパやスイッチがない
- 冷却ファンは不要。S5 と同様、IP 20 保護クラス
- パラメータの割り付けが可能で、さらに診断機能を備えている
- S5 よりも柔軟なスロット割り付けが可能
- 拡張装置および ET 200 リモート I/O 装置で割り込みの起動が可能

S5とS7のモジュールパラメータ割り付けの比較

SIMATIC S5 と SIMATIC S7 のモジュールパラメータの比較を次の表に示します。

SIMATIC S5	SIMATIC S7
	モジュールの配列（ハードウェアコンフィグレーション）は、STEP 7 のハードウェアコンフィグレーション用アプリケーションで行う。
アドレスの設定は DIL スイッチで行う。	アドレスの設定は、STEP 7 のハードウェアコンフィグレーション用アプリケーションで行うか、またはスロット固有。
システム動作は DIL スイッチで設定する。	モジュールパラメータの割り付けは、STEP7のハードウェアコンフィグレーション用アプリケーションで行う。
オペレーション用の CPU パラメータは、システムデータ領域の DB1 / DX0 を介して割り付ける。	CPU パラメータの割り付けは、STEP 7 のハードウェアコンフィグレーション用アプリケーションで行う。
	コンパイルされたコンフィグレーションンデータは、CPU にダウンロードされる。 モジュールパラメータは、スタートアップ時に自動的に転送される。

2.2.1 中央演算装置（CPU）

S7-300 CPU

表 2-1 に、S7-300 CPU の最も重要な性能特性をリストします。 S5 CPU を交換する場合は、この表を基に性能を比較し、最適な CPU を選択することができます。

表 2-1 S7-300 CPU の性能特性

機能	CPU 312 IFM	CPU 313	CPU 314	CPU 314 IFM	CPU 315	CPU 315-2 DP
ワークメモリ（内臓）	6 Kbyte	12 Kbyte	24 Kbyte	24 Kbyte	48 Kbyte	
ロードメモリ						
• 内臓	20 Kbyte の RAM; 20 Kbyte の EEPROM	20 Kbyteの RAM	40 Kbyteの RAM	40 Kbyte の RAM; 40 Kbyte の EEPROM	80 Kbyte の RAM	
• メモリカードによる増設	-	最大 512 Kbyte	最大 512 Kbyte	-	最大 512 Kbyte（CPU で 256 Kbyteまで設定可能）	
プロセスイメージサイズ, 入力と出力	32 byte + 4 オンボード	128 byte	128 byte	124 byte + 4 オンボード	128 byte	
I/O アドレス領域						
• デジタル入力/出力	入力: 128 + 10 オンボード 出力: 128 + 6 オンボード	128	512	入力: 496 + 20 オンボード 出力: 496 + 16 オンボード	1024	
• アナログ入力/出力	32		64	入力: 64 + 4 オンボード 出力: 64 + 1 オンボード	128	
ビットメモリ	1024	2048				
カウンタ	32	64				
タイマ	64	128				
保持データの最大合計量	72 byte		4736 byte	144 byte	4736 byte	
ローカルデータ	合計512 byte 優先度クラス当たり 256 byte	合計1536 byte 優先度クラス当たり 256 byte				
ブロック:						
OB	3	13	13	13	13	14
FB	32	128	128	128	128	128
FC	32	128	128	128	128	128
DB	63	127	127	127	127	127
SFC	25	44	48	48	48	53
SFB	2	7	7	14	7	7

S7-400 CPU

S7-400 の各 CPU はそれぞれ性能特性が異なります。各 CPU の性能特性の比較を表 2-2 に示します。

表 2-2　S7-400 CPU の性能特性

機能	CPU 412-1	CPU 413-1	CPU 413-2 DP	CPU 414-1	CPU 414-2 DP	CPU 416-1	CPU 416-2 DP
ワークメモリ（内臓）	48 Kbyte	72 Kbyte		128 Kbyte	128/384 Kbyte	512 Kbyte	0.8/1.6 Mbyte
ロードメモリ • 内臓 • メモリカードによる増設	8 Kbyte 最大 15 Mbyte			8 Kbyte 最大 15 Mbyte		16 Kbyte 最大 15 Mbyte	
プロセスイメージサイズ, 入力と出力	各 128 byte			各 256 byte		各 512 byte	
I/O アドレス領域 • デジタル入力/出力 最大 • アナログ入力/出力 最大	2 Kbyte 16384 1024			8 Kbyte 65536 4096		16 Kbyte 131072 8192	
ビットメモリ	4096 M 0.0 ~ M 511.7			8192 M 0.0 ~ M 1023.7		16384 M 0.0 ~ M 2047.7	
カウンタ	256 C 0 ~ C 255			256 C 0 ~ C 255		512 C 0 ~ C 511	
タイマ	256 T 0 ~ T 255			256 T 0 ~ T 255		512 T 0 ~ T 511	
ローカルデータ	合計 4 Kbyte			合計 8 Kbyte		合計 16 Kbyte	
ブロック: OB FB FC DB SFB SDB	23 256 256 511 24 512			31 512 1024 1023 24 512		44 2048 2048 4095 24 512	
SFC	55	55	58	55	58	55	58

S7-400 のデータ保持特性

SIMATIC S7-400 の CPU では、タイマ、カウンタ、ビットメモリをバッファリングするのにバックアップバッテリが必要です。

バックアップバッテリ不要の S7-300 保持機能

S7-300 CPU では、タイマ、カウンタ、ビットメモリのバッファリングにバックアップバッテリは必要ありません。同様に、電源異常時にはデータブロックの内容も保持することができます。SIMATIC S7-300 の CPU は、メンテナンスフリーでアドレスやデータのバックアップをとります。これらのアドレスやデータは、電源異常時にも保持されるように指定するパラメータを持っています。

使用できる保持領域のサイズは CPU によって異なります。

保持機能を実行するためのパラメータ割り付け

データ保持領域のサイズは、STEP 7 でハードウェアコンフィグレーションを行うときにパラメータ割り付けダイアログボックスで設定します。

2.2.2 電源モジュール（PS）

電源モジュールは、プログラマブルロジックコントローラ（PLC）ごとに選択できます。

S7-300 の電源モジュール

S7-300 の CPU の電源には 24V 電源（工業用）を使用できます。

S7 のモジュールでは、S7-300 用の電源として以下が用意されています。

モジュール名	出力 電流	出力 電圧	入力 電圧
PS 307	2A	DC24V	AC120 / 230V
PS 307	5A	DC24V	AC120 / 230V
PS 307	10A	DC24V	AC120 / 230V

S7-400 の電源モジュール

モジュール名	出力 電流	出力 電圧	入力 電圧
PS 407 4A	4A 0.5A	DC5V DC24V	AC120 / 230V
PS 407 10A	10A 1A	DC5V DC24V	AC120 / 230V
PS 407 20A	20A 1A	DC5V DC24V	AC120 / 230V
PS 405 4A	4A 0.5A	DC5V DC24V	DC24V
PS 405 10A	10A 1A	DC5V DC24V	DC24V
PS 405 20A	20A 1A	DC5V DC24V	DC24V

詳細については、リファレンスマニュアル **/71/** および **/101/** を参照してください。

2.2.3 インターフェースモジュール（IM）

S5 の一部のインターフェースモジュールは S7 で変更されています。この変更に伴い、ローカルエリア接続も変更されています。S7 では、リモートエリア接続での信号転送に PROFIBUS を使用することを推奨します。

IM モジュールの比較

S5 モジュール	S7-300 モジュール	S7-400 モジュール	説明
IM 305 IM 306 IM 300 / IM 312	IM 365 IM 360 / IM 361	IM 460-0 / IM 461-0 IM 460-1 / IM 461-1	セントラルコンフィグレーション
-	-	IM 460-3 / IM 461-3	リモートエリア（最大 100m）
IM 301 / IM 310	PROFIBUS による接続	PROFIBUS による接続	I/O モジュールとシグナルプリプロセシングモジュールの接続（最大 200m）
IM 304 / IM 314	PROFIBUS による接続	PROFIBUS による接続	リモートエリアでリモート I/O を使用（最大 600m）
		IM 463-2	リモートエリアで S5 拡張デバイスのリモート接続（最大 600m）
IM 307 / IM 317	PROFIBUS による接続	PROFIBUS による接続	光ケーブルによる接続（最大 1500m）
IM 308 / IM 318	PROFIBUS による接続	PROFIBUS による接続	最大 3000m

S7 では、IM 308C の代わりにインターフェースモジュール IM 467 を使用できます。

IM 463-2 インターフェースモジュールは、S5 デジタル/アナログモジュールを S5 増設ラックを介して、IM 314 が装着された S7 マウントラックに接続する場合に使用できます。

接続可能な S5 増設ラック

以下の S5 増設ラックを接続することができます。

- EG 183 増設ユニット
- EG 185 増設ユニット
- ER 701-2
- ER 701-3

2.2.4 コミュニケーションプロセッサ（CP）

次の節では、様々なサブネットで使用できる S5/S7 コミュニケーションプロセッサをリストします。さらに、これらのプロセッサで実行できる機能も示します。

SIMATIC のサブネット

各種のオートメーションレベル（プロセシング、セル、フィールド、アクチュエータセンサの各レベル）で求められる機能に対応するため、SIMATIC では次のサブネットを提供しています。

- **AS インターフェース**

  アクチュエータセンサインターフェース（AS-i）は、処理レベルの低いオートメーションシステムに適した接続システムです。これは主に、バイナリセンサとアクチュエータのネットワーキングに使用されます。このデータ量は 1 スレーブ当たり 4 ビットまでです。

- **MPI**

  マルチポイントインターフェース（MPI）サブネットは、短距離フィールドやセルレベルに適しています。MPI は SIMATIC S7/M7 と C7 で使用されるマルチポイントインターフェースです。これはプログラミング装置用のインターフェースで、少数の CPU 同士を接続したり、少量のデータ（最大 70 バイト）を交換するのに使用します。

- **PROFIBUS**

  PROFIBUS は、メーカーに依存しない、オープンな SIMATIC 通信システムのセル領域およびフィールド領域で使用するネットワークです。中程度のデータ量（約 200 バイト）を高速転送するのに適しています。

- **工業用 Ethernet**

  工業用 Ethernet は、メーカーに依存しない、オープンな SIMATIC 通信システムのプロセシングレベルおよびセルレベルに使用します。大容量のデータを高速転送するのに適しています。

- **ポイントツーポイント接続**

  ポイントツーポイント接続は、従来の意味でのサブネットではありません。この接続は、ポイントツーポイントコミュニケーションプロセッサ（CP）を使って SIMATIC で確立され、2 つの通信パートナー（PLC、スキャナ、PC など）を相互に接続します。

ASインターフェース（SINEC S1）

次の表に、アクチュエータ・センサ（AS）インターフェースに使用するモジュールの概要を示します。

S5 モジュール	S7-300 モジュール	S7-400 モジュール
CP 2433（AS-i 機能） CP 2430（AS-i 機能）	CP 342-2（AS-i 機能）	-

MPI（SINEC L1）

S5 での SINEC L1 経由の通信は、S7 では MPI を使ってグローバルデータ通信に変換されています。

S7-300 および S7-400 のすべての CPU、プログラミング装置、オペレータパネルには MPI インターフェースが装備されています。

PROFIBUS（SINEC L2）

次の表に、PROFIBUS との通信に使用できるモジュールとこれらのモジュールでサポートされる機能に関する概要を示します。

S5 モジュール	S7-300 モジュール	S7-400 モジュール
CP 5431 (FMS, FDL, DP) CPU 95U (FDL, DP *))	CP 342-5 (S7 ファンクション, FDL, DP) CP 343-5 (S7 ファンクション, FDL, FMS)	CP 443-5 Ext. (S7 ファンクション, FDL, DP) CP 443-5 Basic (S7 ファンクション, FDL, FMS)
IM 308-B/C (DP)	CPU 315-2 DP (DP)	CPU 413-2 DP (DP) CPU 414-2 DP (DP) CPU 416-2 DP (DP) IM 467 (DP)

*) 注文した装置によって異なる

工業用 EthernetSINEC H1）

次の表に、工業用 Ethernet との通信に使用できるモジュールとこれらのモジュールでサポートされる機能に関する概要を示します。

S5 モジュール	S7-300 モジュール	S7-400 モジュール
CP 1430 TF (ISO トランスポート)	CP 343-1 (S7 ファンクション, ISO トランスポート)	CP 443-1 (S7 ファンクション, ISO トランスポート)
CP 1430 TCP (ISO on TCP)	CP 343-1 TCP (S7 ファンクション, ISO on TCP)	CP 443-1 TCP (S7 ファンクション, ISO on TCP)

ポイントツーポイント接続

次の表に、ポイントツーポイント接続に使用できるモジュールとこれらのモジュールでサポートされる機能に関する概要を示します。

S5 モジュール	S7-300 モジュール	S7-400 モジュール
CP 521 (3964 (R), ASCII) CP 523 (3964 (R), ASCII)	CP 340-RS 232C (3964 (R), ASCII) CP 340-20 mA (3964 (R), ASCII) CP 340-RS 422/485 (3964 (R), ASCII)	CP 441-1 (3964 (R), RK512, ASCII)
CP 544 (3964 (R), RK 512, ASCII)	-	
CP 524/525 (3964 (R), RK 512, ASCII, ロード可能なその他の特殊ドライバ CP 544 B (3964 (R), RK 512, ASCII, ロード可能なその他の特殊ドライバ	-	CP 441-2 (3964 (R), RK512, ASCII, ロード可能なその他の特殊ドライバ

2.2.5 ファンクションモジュール（FM）

SIMATIC S5 の一部の IP および WF モジュールは、専用アダプタケースを使えば S7-400 で使用できます。これ以外については、必要な機能を使用できるようにするための新しいファンクションモジュールが S7 に用意されています。

次の表に、S5 と S7 のシグナルプリプロセシングモジュールの概要および比較を示します。

表 2-3 シグナルプリプロセシングモジュール S5 と S7 の比較

S5 モジュール	アダプタケーシング	S7 モジュール	説明
IP 240	あり	FM 451（制限あり）	カウンタ、位置検出、位置決めモジュール
IP 241	なし	FM 451 / FM 452（制限あり）	デジタル位置検出モジュール
IP 242A	なし	なし	カウンタモジュール
IP 242B	あり	なし	カウンタモジュール
IP 244	あり	FM 455	コントローラモジュール
IP 246I/A	あり	FM 354 / FM 357 / FM 453	可変速ドライブ用位置決めモジュール
IP 247	あり	FM 353 / FM 357 / FM 453	ステッピングモータ用位置決めモジュール
IP 252	なし	FM 455（制限あり）	クローズドループコントロールモジュール
IP 260	なし	FM 355（制限あり）	クローズドループコントロールモジュール
IP 261	なし	なし	位置決めモジュール
IP 281	なし	FM 350-1 / FM 450-1	カウンタモジュール
IP 288	なし	FM 451 / FM 452	高速/低速制御およびカムコントロール用位置決めモジュール
WF 705	あり	FM 451（制限あり）	位置検出モジュール
WF 706	なし	FM 451（制限あり）	位置決めおよびカウンタモジュール
WF 707	なし	FM 452（制限あり）	カムコントロール
WF 721	あり	FM 354（アセンブリ技術が原因で制限あり）	位置決めモジュール

表 2-3　　シグナルプリプロセシングモジュール S5 と S7 の比較、続き

S5 モジュール	アダプタケーシング	S7 モジュール	説明
WF 723A	あり	FM 453	位置決めモジュール
WF 723 B	あり	FM 357（アセンブリ技術が原因で制限あり）	位置決めモジュール
WF 723 C	あり	なし	位置決めモジュール
-	-	FM 456-4	アプリケーションモジュール（M7-FM）
-	-	SINUMERIK FM-NC	数値制御
-	-	FM STEPDRIVE	ステッピングモータ制御
-	-	SIMOSTEP	ステッピングモータ

2.2.6 シグナルモジュール（SM）

SIMATIC S7 のシグナルモジュールは、S5 の入力/出力モジュールと機能上の互換性があります。ただし、S7 シグナルモジュールは、シグナルモジュールの機能に加えて、パラメータ割り付けが可能で、しかも診断機能を備えています。

パラメータ割り付けが可能なシグナルモジュール

パラメータ割り付けが可能な S7 デジタル入力モジュールにより、エッジ変化でハードウェア割り込みを起動するチャンネルを指定できます（ハードウェアコンフィグレーション用の STEP 7 アプリケーションを使用）。

アナログ入力モジュールの入力レンジには、STEP 7 で簡単にパラメータを割り付けることができます。

診断機能付きのシグナルモジュール

診断機能付きモジュールにより、外部エラー（断線やモジュール以外の短絡）および内部エラー（RAM エラーやモジュールの短絡）を検出することができます。

診断イベントは、コントローラにより次の 2 つの方法で処理されます。

- 診断割り込みを起動する。これは、ユーザプログラムの該当するオーガニゼーションブロック（OB）に通知され、それからサイクリックプログラムに割り込みをかけます。
- CPU の診断バッファにエントリを作成する。このエントリはプログラミング装置またはオペレータインターフェース装置で読み取ることができます。

次の表に、S7 のシグナルモジュールをリストします。

表 2-4 SIMATIC S7-300 のシグナルモジュール

<table>
<tr><th>DI
（SM 321）</th><th>DO
（SM 322）</th><th>AI
（SM 331）</th><th>AO
（SM 332）</th></tr>
<tr><td>32 x DC24V</td><td>32 x DC24V/0.5 A</td><td>8 x 12 bit</td><td rowspan="2">2 x 12 bit</td></tr>
<tr><td>16 x DC24V</td><td>16 x DC24V/0.5 A</td><td>2 x 12 bit</td></tr>
<tr><td>16 x DC24V
ハードウェア割り込みと診断割り込みあり</td><td>8 x DC24V/0.5 A
診断割り込み機能あり</td><td>Ex: 4 x 15 bit</td><td rowspan="2">Ex: 4 x 15 bit</td></tr>
<tr><td>16 x DC24V
Mリーディング</td><td>8 x DC24V/2 A</td><td>Ex: 12 x 15 bit</td></tr>
<tr><td>8 x AC120/230V</td><td>8 x AC120/230V /
2 A</td><td colspan="2" rowspan="3">AI 4/AO 2 X 8/8 bit（SM 334）</td></tr>
<tr><td rowspan="2">Ex: 4 x DC24V</td><td>Ex: 4 x DC15V/
20mA</td></tr>
<tr><td>Ex: 4 x DC24V/
20mA</td></tr>
</table>

表 2-5　　SIMATIC S7-400 のシグナルモジュール

DI （SM 421）	DO （SM 422）	AI （SM 431）	AO （SM 432）
32 x DC24V	32 x DC24V/0.5 A	8 x 13 bit	8 x 13 bit
16 x UC24/60V ハードウェア割り込みと診断割り込みあり	16 x DC24V/2 A	8 x 14 bit （温度測定用）	
16 x UC120/230V	16 x AC120/230V /5 A	8 x 14 bit	
32 x UC120V	16 x AC120/230V /2 A	16 x 16 bit	
	16 x UC0/230V/ Rel. 5 A		

2.2.7　シミュレーションモジュール（S7-300）

S7-300 には、プログラムをテストするためのシミュレーションモジュール SM 374 が用意されています。

このシミュレーションモジュールには以下の機能があります。

- シミュレーション能力
  - 入力 16 点
  - 出力 16 点
  - 入力 8 点と出力 8 点（それぞれ同じ初期アドレス）
- 機能はドライバーで設定できる
- 入力/出力のシミュレーションの状態を表示できる

2.3 リモート I/O 装置

SIMATIC S5 には ET 200 システムのリモート I/O 装置用モジュールがありますが、これは SIMATIC S7 でも引き続き使用できます。

さらに、有効レンジが拡張された新しい ET 200 モジュールも用意されています。

DP マスタ

以下のモジュールは、リモート I/O システムで DP マスタとして使用できます。

- CPU 315-2 DP または CP 342-5 を DP マスタとして搭載した S7-300
- CPU 413-2 DP / 414-2 DP / 416-2DP または CP 443-5 を DP マスタとして搭載した S7-400

DP スレーブ

以下の装置は、リモート I/O システムで DP スレーブとして使用できます。

- リモート I/O 装置。たとえば ET 200B、ET 200C、ET 200M、ET 200X（最大 12 Mbps）、および ET 200U、ET 200L（最大 1.5 Mbps）
- プログラマブルロジックコントローラ
  - IM 308-C を DP スレーブとして搭載した S5-115U、S5-135U、または S5-155U
  - DP スレーブインターフェースを装備したS5-95U（最大 1.5 Mbps）
  - CPU 315-2 DP または CP 342-5 を DP スレーブとして搭載したS7-300
  - CP 443-5 を DP スレーブとして搭載した S7-400
- DP/AS-i リンクによる、アクチュエータ/センサインターフェースへのインターフェース
- マシン方式のオペレータコントロールおよびモニタリングに使用するテキスト表示とオペレータパネル
- MOBY 識別システム
- 低電圧スイッチング装置
- Siemens 製または他社製のフィールド装置（ドライブやバルブアイランドなど）

FMS マスタ

以下のモジュールは **FMS マスタ**として使用できます。

- CP 343-5 を FMS マスタとして搭載した S7-300
- CP 443-5 Basic を FMS マスタとして搭載した S7-400

FMS スレーブ

FMS スレーブとして機能する装置として、ET 200U や SIMOCODE モータ保護制御装置があります。

詳細については、該当するマニュアルまたは Siemens カタログ CA01 を参照してください。

2.4 通信

サービスおよびサブネット

SIMATIC S7 での通信は、様々なサービスを提供する各種サブネットを基本としています。

サービス	S7 通信機能 （S7 ファンクション）		
	ISOトランスポート ISO-on-TCP	FDL（SDA） FMS DP	GD
サブネット	工業用 Ethernet	PROFIBUS	MPI

SIMATIC で使用される通信サービスを次にまとめます。

S7 ファンクション

S7 ファンクションにより、S7/M7 CPU、S7 OP/OS、および PC 間の通信サービスを提供します。これらの S7 ファンクションは各 SIMATIC S7/M7 装置に組み込まれています。S7 ファンクションは ISO アプリケーション層のサービスに対応するため、どのサブネットからも独立しており、すべてのサブネット（MPI、PROFIBUS、工業用 Ethernet）で使用することができます。

ISO トランスポート

これらの機能は、SIMATIC S7 から SIMATIC S5 へのデータ転送を確実に実行するために使用されます。

これは、工業用 Ethernet の ISO 参照モデルに基づく ISO トランスポート層で開放型通信を行うことにより、中程度のデータ量（最大 240 バイト）を転送するのに使用されます。

ISO on TCP

これらの機能は、SIMATIC S7 から SIMATIC S5 へのデータ転送を確実に実行するために使用されます。

これは、工業用 Ethernet の ISO 参照モデルに基づく ISO トランスポート層で TCP/IP プロトコルに従って開放型通信を行うことにより、中程度のデータ量（最大 240 バイト）を転送するのに使用されます。

ISO-on-TCP サービスには、拡張 RFC1006 規格が必要です。

FDL（SDA）

これらの機能は、SIMATIC S7 から SIMATIC S5 へのデータ転送を確実に実行するために使用されます。

これは、工業用 Ethernet の ISO 参照モデルに基づくフィールドバスデータリンク（FDL）層で開放型通信を行うことにより、中程度のデータ量（最大 240 バイト）を転送するのに使用されます。

FMS

PROFIBUS FMS（Fieldbus Message Specification）は、スタティック FMS 接続を介して構造化データ（FMS 変数）を転送するためのサービスを提供します。

FMS サービスは、ISO 参照モデルの第 7 層に分類できます。欧州規格EN 50170 Vol. 2 PROFIBUS に適合し、構造化データ（変数）転送のサービスを提供します。

DP

PROFIBUS DP サービスにより、リモート I/O 装置と透過的に通信を行えます。このリモート I/O 装置は、制御プログラムによって中央 I/O 装置と同じ方法でアドレス指定できます。

GD

グローバルデータ通信は、S7-300/400 CPU のオペレーティングシステムに組み込まれている単純な通信オプションです。

GD 通信により、マルチポイントインターフェースを介して CPU 間のデータ交換を周期的に実行できます。S7-400 では、イベント駆動のデータ交換も実行できます。

2.4.1 ユーザプログラムとのインターフェース

ユーザプログラムとの通信インターフェースは以下のブロックで構成されます。

- SFC（接続コンフィグレーションなし）
- SFB（接続コンフィグレーションなし）（S7-400 のみ）
- ロード可能な FC / FB

S5 ハンドリングブロックに代わって上記のブロックが使用されます。 どちらも機能は似ていますが、新ブロックには STEP 7 が使用されています。通信を確立するには、ハンドリングファンクションを使って、該当する S5 プログラムを新しいブロックに適合させなければなりません。

ネットワーク	サービス	S5 ユーザプログラムのインターフェース	S7 ユーザプログラムのインターフェース
ポイントツーポイント接続	-	ハンドリングブロック*	S7-300: ロード可能FB S7-400: ロード可能SFB
PROFIBUS	FDLPLC-PLC） Free Layer 2 FMS	ハンドリングブロック* ハンドリングブロック* ハンドリングブロック*	ロード可能FC - ロード可能FB
工業用Ethernet	ISO 4 ISO 4 + AP STF MAP	ハンドリングブロック* ハンドリングブロック* ハンドリングブロック*+ロード可能 FB ハンドリングブロック*+ロード可能 FB	ロード可能FC - - ロード可能FB

* CPU に応じて内臓またはロード可能

2.5 オペレータコントロールおよびモニタリング

はじめに

次のセクションでは、SIMATIC HMI（Human Machine Interface、旧名称はCOROS）オペレータパネルを SIMATIC S7 でどの程度まで使用できるのかについて、その概要を説明します。

オペレータパネル

SIMATIC HMI オペレータパネルは、SIMATIC S5、SIMATIC S7、SIMATIC TI、その他のコントローラのオペレータ制御とモニタリング機能を提供します。

STEP 5

一般に、標準ファンクションブロック（接続されたオペレータパネルに応じて呼び出される）は、プログラマブルコントローラで **SIMATIC OP を SIMATIC S5** に接続する場合に必要になります。

以下のオペレータパネル（OP）は S5 で使用できます。

- TD17、OP5/A1、OP7/PP、OP7/DP-12、OP15/x1、OP17/PP、OP17/DP-12
- OP25、OP35、OP37、TP37

STEP 7

SIMATIC OP と SIMATIC S7/M7 に接続する場合、MPI ノードとしての PPI、MPI、PROFIBUS を区別しなければなりません。

PPI または MPI 接続は、CPU のプログラミング装置インターフェースを介して実行されます。このとき、SIMATIC OP は SIMATIC S7/M7 の通信サービス（S7 ファンクション）を使用します。つまり、標準ファンクションブロックは必要ありません。

一方、SIMATIC OP から SIMATIC S7/M7 への PROFIBUS 接続では、S7 ファンクションを使って通信を行います。つまり、この場合も標準ファンクションブロックは必要ありません（SIMATIC S5 と PROFIBUS 接続する場合、SIMATIC OP は「アクティブなノード」で、PROFIBUS-DP スレーブではありません）。MPI 接続の場合と同数のノードが使用されます。

S7 では以下のオペレータパネル（OP）を使用できます。

- TD17、OP3、OP5/A2、OP7/DP、OP7/DP-12、OP15/x2、OP17/DP、OP17/DP-12
- OP25、OP35、OP37、TP37

SIMATIC OP には次の制限があります。

- OP3: 最大 2 接続
- OP5/15/25: 最大 4 接続
- TD17, OP7/17: 最大 4 接続
- OP35: 最大 6 接続
- OP37, TP37: 最大 8 接続

コンフィグレーション

SIMATIC ProTool および SIMATIC ProTool/Lite は、オペレータパネルをコンフィグレーションするための最新ツールです。SIMATIC ProTool はすべての装置のコンフィグレーションに使用できるのに対し、SIMATIC ProTool/Lite はライン表示のオペレータパネルにのみ使用できます。

SIMATIC STEP 7 への組み込み

ProTool は、SIMATIC STEP 7 コンフィグレーションソフトウェアに組み込むことができるために、シンボルテーブルや通信パラメータなど、制御コンフィグレーションに使用されるコンフィグレーションデータに直接アクセスできます。これにより、時間と経費を節約できるだけでなく、データエントリの重複が原因のエラーを防ぐことができます。

表 2-6　　オペレータインターフェース装置のコンフィグレーション

装置	コンフィグレーションツール
ライン表示 OP（TD17, OP3, OP5, OP7, OP15, OP17）	ProTool/Lite または ProTool
グラフィック表示 OP（OP25, OP35, OP37, TP37）	ProTool

WinCC

WinCC は、単一ターミナルまたは複数ターミナル（クライアント・サーバ構成）のシステムに使用できます。

WinCC は、生産オートメーションおよびプロセスオートメーションに使用されるビジュアル化制御タスクとプロセス制御タスクのソリューションを作成するためのシステムです。これは、すべてのビジネスセクタおよびテクノロジとの互換性をもっています。これにより、グラフィックとメッセージの表示、情報のアーカイブ、工業用途の記録保存に適したファンクションモジュールを提供します。ハードウェア接続がパワフルで効率的、表示更新がスピーディー、データのアーカイブが確実なため、フレキシビリティに富み、高いアベイラビリティが実現します。

WinCC は、これらのシステムファンクションに加えて、ユーザ固有ソリューションを作成するためのオープンインターフェースを提供します。これにより、複合的な全社規模のオートメーションソリューションに WinCC を統合することができます。さらに、統合機能により ODBC や SQL などの標準インターフェース経由でデータアーカイブへアクセスしたり、OLD2.0 および OLE カスタムコントロール（OCX）経由によりオブジェクトとドキュメントを統合することもできます。これらのメカニズムにより、WinCC は Windows アプリケーションの効果的な通信パートナーとなります。

WinCC は、32 ビットオペレーティングシステムの MS Windows 95 または MS Windows NT を基本にしています。どちらもプリエンプティブマルチタスキングのため、プロセスイベントに迅速に応答し、データロスに対して高レベルのセキュリティを提供します。Windows NT ではさらにセキュリティ機能が追加され、WinCC マルチターミナルシステムでサーバオペレーションの基盤として機能します。WinCC ソフトウェア自体は、最新のオブジェクト指向ソフトウェア技術を使って開発された32 ビットアプリケーションです。

ソフトウェア 3

3.1 一般的な動作原理

概要

SIMATIC S7/M7/C7 のコンフィグレーション/プログラミング用ソフトウェアは、最新の人間工学的コンセプトに従って設計されているため、非常にわかりやすくできています。

3.1.1 インストールの条件

オペレーティングシステム

Microsoft Windows 95

標準的なハードウェア

以下の仕様および装置を備えたプログラミング装置または PC:

- 80486 以上のプロセッサ
- 最低で 16Mバイトの RAM（32Mバイトを推奨）
- VGA モニタ、または Windows 95 でサポートされるタイプのモニタ
- Microsoft Windows 95 に対応するキーボード、およびマウス（オプションだが推奨）

ハードディスクの空き容量

ハードディスクには以下に示す空き容量が必要です。

- 標準パッケージで1言語をインストールする場合は、ハードディスクに 105 Mbyte が必要です。インストールする標準ソフトウェアの容量によって正確な必要容量が異なります。
- STEP 7 では、スワップファイルの保存に必要な総メモリ容量は約 64 Mbyteです。たとえば、32 Mbyte の RAM がある場合、さらに 32 Mbyte の仮想メモリが必要です。
- ユーザデータには約 50 Mbyte を確保する必要があります。
- セットアップファイル用に、ハードディスクに最低 1 Mbyte が必要です（インストールが完了すると、セットアップファイルは削除されます）。

3.1.2 STEP 7 ソフトウェアのインストール

概要

STEP 7 には、インストールを自動的に実行するセットアッププログラムが入っています。画面にユーザプロンプトが表示され、インストール手順全体をステップバイステップ方式で示します。

認証

STEP 7 プログラミングソフトウェアを使用するには、製品固有のユーザ認証が必要です。この方法で保護されるソフトウェアは、プログラムまたはソフトウェアパッケージに必要な認証が各プログラミング装置/PC のハードディスクに格納されていなければ、使用することはできません。

この認証を取得するには、パッケージに同梱されているコピー禁止認証ディスケットが必要です。このディスケットには、STEP 7 の表示、インストール、アンインストールに必要なプログラム AUTHORS も格納されています。

この認証の転送方法および削除方法については、「*User Manual* **/231/**」で説明されています。

注

Siemens プログラミング装置（PG 740 など）には、インストール可能な STEP 7 ソフトウェアがあらかじめハードディスクに格納されています。

STEP 7 のインストールの詳細については、「*User Manual* **/231/**」を参照してください。

3.1.3 STEP 7 ソフトウェアの開始

開始

Windows 95/NT を起動すると、Windows ユーザインターフェースに SIMATIC 、マネージャのアイコンが表示されます。このアイコンから、STEP 7 ソフトウェアへアクセスします。

STEP 7 を起動する最も迅速な方法は、[SIMATIC Manager] アイコンをダブルクリックする方法です。このアイコンにより、SIMATIC マネージャのウィンドウが開きます。このウィンドウから、ユーザがインストールした標準システム、オプションソフトウェア、ファンクションにアクセスできます。

このアイコンをダブルクリックする以外の方法として、Windows 95/NT ツールバーの [開始] ボタンをクリックして SIMATIC マネージャを起動することもできます。このメニューは、“Simatic/STEP 7” にあります。

SIMATIC マネージャ

SIMATIC マネージャは、コンフィグレーションとプログラミングに使用する初期ウィンドウです。このウィンドウでは以下を実行できます。

- プロジェクトのセットアップ
- パラメータの設定とハードウェアへの割り付け
- 通信接続のコンフィグレーション
- プログラムの作成
- プログラムをテストし、実行する

各ファンクションへのアクセスは、オブジェクト指向ベースの直感的な方法で行え、すぐに覚えることができます。

SIMATIC マネージャは次の方法で使用できます。

- オフライン（コントローラに接続されていない）
- オンライン（コントローラに接続されている）

（上記を実行するには、所定の安全上のガイドラインに必ず従ってください）

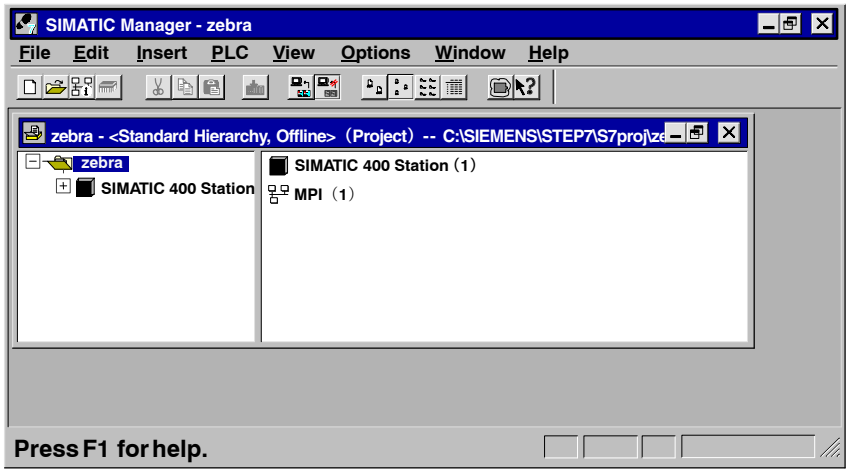

図 3-1 SIMATIC マネージャでプロジェクトが開いている状態

3.2 S7 プロジェクトの構造

定義

プロジェクトには、オートメーションソリューションのすべてのデータとプログラムが含まれています。プロジェクトの目的は、オートメーションソリューション用に作成されたデータやプログラムを系統的に保存することです。

STEP 5 のプロジェクト

「プロジェクト」という言葉は STEP 5 ですでにご存じのことでしょう。STEP 5 のプロジェクトでは、1ユーザプログラムに作成された STEP 5 ファイルはすべて 1 つのプロジェクトファイルに格納されます。

このプロジェクトファイルには、パラメータ設定やカタログ名、あるいはファイル名など、ユーザプログラムの編集やメンテナンスを効果的に実行するための情報が格納されています。

STEP 7 のプロジェクト

STEP 7 のプロジェクトには、使用する CPU の数やそれらの接続方法に関係なく、オートメーションソリューションに必要なプログラムとデータがすべて含まれています。このため、プロジェクトは、特定のプログラマブルモジュールに使用されるユーザプログラムに限定されるのではなく、多数のプログラマブルモジュールで使用される複数のユーザプログラムを格納することができます。これらのユーザプログラムは、すべてまとめて共通のプロジェクト名で保存されます。

注

STEP 7 では STEP 5 と同様に、1個の CPU を対象とした単純なユーザプログラムを作成することができます。この場合、プロジェクトは 1 個の CPU に限定されます。

次の節では、STEP 7 でユーザが作成するユーザプログラムとデータのディレクトリ構造を説明します。

プロジェクトのコンポーネント

STEP 7 のプロジェクトは、基本的に図 3-2 に示すオブジェクトで構成されます。これらのオブジェクトを以下にリストし、説明します。

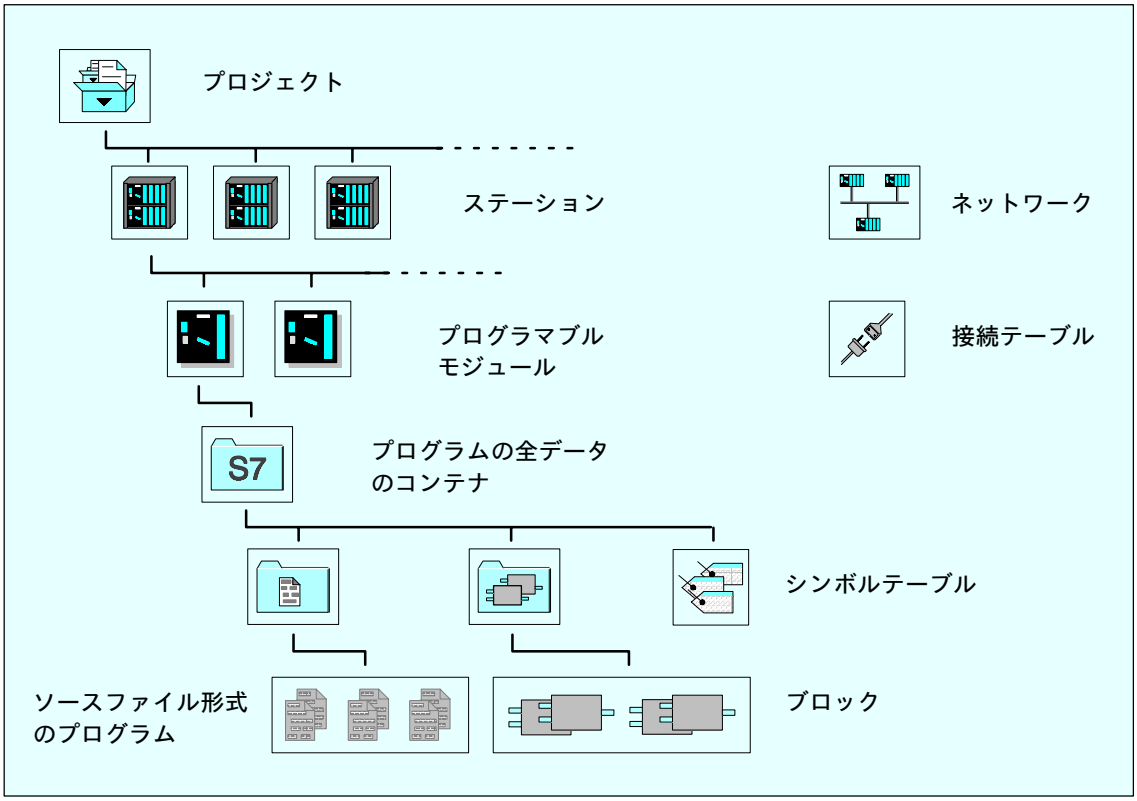

図 3-2　　STEP 7 プロジェクトの基本オブジェクトの階層図

ネットワーク

「ネットワーク」オブジェクトには、MPI や PROFIBUS などのサブネットの定義プロパティが含まれています。ステーションやステーション内のコミュニケーションモジュールをネットワークに割り付けると、STEP 7 で通信パートナーの整合性をチェックすることができます。

ステーション

ステーションは、プログラマブルコントローラとこのプログラマブルコントローラに使用されているすべてのラックの構造のことです。DP インターフェースを装備したモジュールがステーションに含まれている場合、マスタシステム全体（つまりマスタシステムに属する DP スレーブ）もこのステーションの一部になります。

ステーションは、1つまたは複数のプログラマブルモジュール（たとえば CPU など）で構成されます。

ハードウェア

ハードウェアは、ステーションのコンフィグレーションデータおよびパラメータを含むオブジェクトです。ステーションのコンフィグレーションデータおよびパラメータはシステムデータブロック（SDB）に格納されます。

プログラマブルモジュール

プログラマブルモジュールは、他のモジュールとは異なり、ユーザプログラムが格納されています。プログラマブルモジュールのフォルダ（STEP 7 ではコンテナと呼ぶ）には、モジュールのプログラムに属するすべてのデータが格納されています。このプログラムには次のようなものがあります。

- ソースファイル形式のプログラム（テキストエディタで作成）

  ソースプログラムがコンパイルされると、「ブロック」コンテナに実行可能ブロックが作成されます。
- プログラマブルモジュールにロードされるブロック
- シンボルテーブル

接続テーブル

接続テーブルは、ステーション内のプログラマブルモジュール（CPU など）のすべての接続を表します。接続は、2 つのノード間の通信プロパティを定義し、接続 ID で識別されます。標準コミュニケーションブロック（STEP 5 のハンドリングブロックに類似）を使ってイベント制御通信をプログラミングする場合、この接続 ID のみが必要です。

ソースファイル

S7 プログラミングでは、ソースファイルを基にブロックが作成されます。ソースファイルを S7 CPU にダウンロードすることはできません。

ブロック

ブロックは、ユーザプログラムとは別個のパートで、その機能、構造、使用法によって区別されます。ブロックは、S7 CPU にダウンロードできます。

「ブロック」コンテナには、実行可能ブロックに加えて変数テーブルも格納されています。

シンボルテーブル

シンボルテーブルには、シンボル名（入力、出力、ビットメモリ、ブロックなどの）の割り付けが示されます。

3.3 SIMATIC マネージャによるプロジェクトの編集

3.3.1 プロジェクトの作成

新規プロジェクト

プロジェクトを作成するには、次の手順に従います。

1. SIMATIC マネージャでメニューコマンド [**ファイル▸新規作成**] を選択します。
2. [新規作成] ダイアログボックスでオプション [プロジェクトの新規作成] を選択します。
3. プロジェクトの名前を入力し、[OK] をクリックして入力内容を確定します。

その他の方法

プロジェクトを編集する際、ほとんどのタスクは任意の順序で実行できます。プロジェクトを作成したら、次のいずれかの方法を選択できます。

- ハードウェアをコンフィグレーションし、そのハードウェア用のソフトウェアを作成する
- コンフィグレーションするハードウェアに関係なくソフトウェアを作成する。ステーションのハードウェアコンフィグレーションは、プログラムを入力してからでもかまいません。

表 3-1　その他の方法

その 1	**その 2**
ハードウェアイをコンフィグレーションする（第 3 章 4 節も参照）	**ソフトウェアを作成する**
ハードウェアをコンフィグレーションする （第 3 章 4 節を参照）	
コンフィグレーションが完了すると、ソフトウェアの作成に必要な “S7 Program” コンテナが挿入され、使用できるようになっている	必要なソフトウェアコンテナ（S7 Programs）をプロジェクトに挿入する （第 3 章 6 節を参照）
プログラマブルモジュールのソフトウェアを作成する （第 3 章 6 節を参照）	プログラマブルモジュールのソフトウェアを作成する （第 3 章 6 節を参照）
	ハードウェアをコンフィグレーションする （第 3 章 4 節を参照）
	ハードウェアのコンフィグレーションが完了したら、S7 プログラムを CPU にリンクする

ハードウェアのコンフィグレーションを行わずに作成したプログラムのダウンロード方法およびテスト方法については、「*User Manual* **/231/**」を参照してください。

3.3.2 プロジェクトの保存

概要

プロジェクトをバックアップする場合、そのプロジェクトのコピーを別名で保存したり、アーカイブすることができます。

名前を付けて保存

別の名前でプロジェクトを保存するには、次の手順に従います。

1. プロジェクトを開きます。
2. メニューコマンド [**ファイル▸名前を付けて保存**] を選択します。
   [名前を付けて保存] ダイアログボックスが表示されます。
3. 保存時に一貫性チェックを行うのかどうかを選択し、[OK] をクリックしてダイアログボックスを閉じます。[名前を付けて保存] ダイアログボックスが表示されます。
4. [保存する場所] で、プロジェクトの保存先となるディレクトリを選択します。
5. [ファイル名] 欄で、アスタリスク（*）の代わりにファイル名を入力します。ファイルの拡張子は変更しないでください。
6. [OK] をクリックしてダイアログボックスを閉じます。

選択したドライブに十分な空き容量があることを確認してください。たとえば、プロジェクトは大きくてディスケットには収まらないため、バックアップ先にディスクドライブを選択することはお勧めできません。この場合、プロジェクトをアーカイブしてから、複数のディスケットに保存しなければなりません。アーカイブされたデータは、分割して複数のディスケットに保存することができます。

アーカイブ

各プロジェクトまたはライブラリは、ハードディスクやリムーバブルメディア（ディスケット）にあるアーカイブファイルに圧縮形式で保存することができます。

アーカイブしたプロジェクトやライブラリのコンポーネントにアクセスするには、そのプロジェクトを解凍しなければなりません。解凍方法については、「*User Manual* **/231/**」で説明します。

3.4 STEP 7 によるハードウェアのコンフィグレーション

SIMATIC S5 では、ソフトウェアを使ってハードウェアをコンフィグレーションするオプションはありませんでした。S7 では、アドレス指定、モジュールへのパラメータの割り付け、通信のコンフィグレーションは、STEP 7 アプリケーションで行います。STEP 7 アプリケーションを使用すると、プログラミング装置からコンフィグレーションやパラメータ割り付けを実行できるので、モジュール側の設定は一切必要ありません。

前提条件

ハードウェアをコンフィグレーションするには、プロジェクトをあらかじめ作成しておかなければなりません

ステーションの挿入

プロジェクトに新しいステーションを作成するには、プロジェクトを開いてプロジェクトウィンドウを表示します（まだ開いていない場合）。

1. プロジェクトを選択します。
2. メニューコマンド [**挿入 ▸ ステーション**] を選択して、目的のハードウェアに対してオブジェクトを作成します。

以下のオプションをサブメニューで選択できます。

- SIMATIC 300 ステーション
- SIMATIC 400 ステーション
- PC/プログラミング装置
- SIMATIC S5
- その他のステーション
  （つまり、SIMATIC S7/M7 または SIMATIC S5 以外）

PC/プログラミング装置、SIMATIC S5、その他のステーションは、通信リンクのコンフィグレーションのみが可能です。S5 ステーションでコンフィグレーションおよびプログラミングを行うことはできません。

プロジェクトウィンドウのプロジェクトアイコンの下にステーションが表示されない場合は、このアイコンの前にある “+” 記号をクリックしてください。

ハードウェアのコンフィグレーション

ハードウェアをコンフィグレーションするには、次の手順に従います。

1. 新しく挿入したステーションをクリックします。このステーションには “Hardware” オブジェクトが格納されています。
2. “Hardware” オブジェクトを開きます。[HWConfig] ウィンドウが表示されます。
3. [ハードウェアコンフィグレーション] ウィンドウで、ステーションの構造を計画します。このとき、モジュールのカタログが役に立ちます。このカタログが表示されていない場合は、メニューコマンド[**表示 ▸ カタログ**] を選択して表示します。
4. モジュールカタログのラックを空のウィンドウに挿入します。次にモジュールを選択し、ラックスロットに挿入します。各ステーションに 1 個以上の CPU をコンフィグレーションする必要があります。この手順の実行中、ユーザが作成したエントリは HWConfig によって自動的にチェックされます。

ハードウェアのコンフィグレーションについては、「*User Manual* **/231/**」を参照してください。

コンフィグレーションの結果

ハードウェアコンフィグレーションを保存して終了すると、コンフィグレーションを行った各 CPU に対して S7 プログラムと接続テーブル（“Connections” オブジェクト）が自動的に作成されます。S7 プログラムには、ソフトウェアコンテナである “Source Files” オブジェクトと “Blocks” オブジェクト、さらにはシンボルテーブルが含まれています。

“Blocks” コンテナには、OB1 のオブジェクト、“System Data” オブジェクト、コンパイル済みのコンフィグレーションデータが格納されています。

3.5 接続テーブルでの接続のコンフィグレーション

S5 では、COM NCM を使って接続のコンフィグレーションを行います。コミュニケーションプロセッサ（CP）ごとに COM パッケージが用意されています。S7 では、コンフィグレーションテーブルですべての接続をコンフィグレーションします。

概要

ユーザプログラムで SFB 通信機能を使用するには、まず最初に接続をコンフィグレーションしなければなりません。

接続では以下を指定します。

- S7 プロジェクトで使用する通信パートナー
- 確立する接続のタイプ（S7 接続や FDL 接続など）
- 接続の確立が能動型または受動型なのか、動作モードメッセージを送信するかどうかなどの特殊プロパティ

接続をコンフィグレーションするとき、接続ごとに一意のローカル識別子（ローカル ID）が発行されます。通信パートナーの割り付けには、このローカル ID のみが必要です。

接続の端点となる CPU にはそれぞれ独自の接続テーブルがあります。

特殊機能

通信パートナー同士が S7-400 ステーションの場合、接続の両端に対してローカル ID が自動的に発行されます。S7-400 ステーションを S7-300 ステーションと接続する場合、ローカル ID は 1 つだけ生成されます。

コンフィグレーションデータのロード

S7 ステーションにおける接続端点のローカルコンフィグレーションデータは、各ターゲットステーションへ別々にダウンロードしなければなりません。

（空の）接続テーブル（“Connections” オブジェクト）が、CPU ごとに自動的に作成されます。この接続テーブルにより、ネットワーク内の CPU 同士の通信リンクを定義します。接続テーブルが開くと、ウィンドウが現れ、プログラマブルモジュール間の接続を定義するテーブルが表示されます（接続の定義については、「*User Manual* **/231/**」を参照）。

例: S5 装置への接続

この例は、SIMATIC S5 ステーションへの接続をコンフィグレーションする方法を示しています。この例では、SIMATIC 400 ステーションがすでにプロジェクトに挿入されていることが前提です。

- SIMATIC S5 ステーションをプロジェクトに挿入し、ステーションのプロパティを設定する。
- S7 ステーションの接続テーブルを開き、メニューコマンド [**挿入▸接続**] を選択して接続を挿入します。ダイアログボックスが表示されるので、ここで通信パートナー（SIMATIC S5 ステーション）と接続タイプを入力します。
- 上記を入力したら、接続テーブルに接続が表示されます。この接続のプロパティを、S5 ステーションの対応する COM NCM に入力する必要があります。

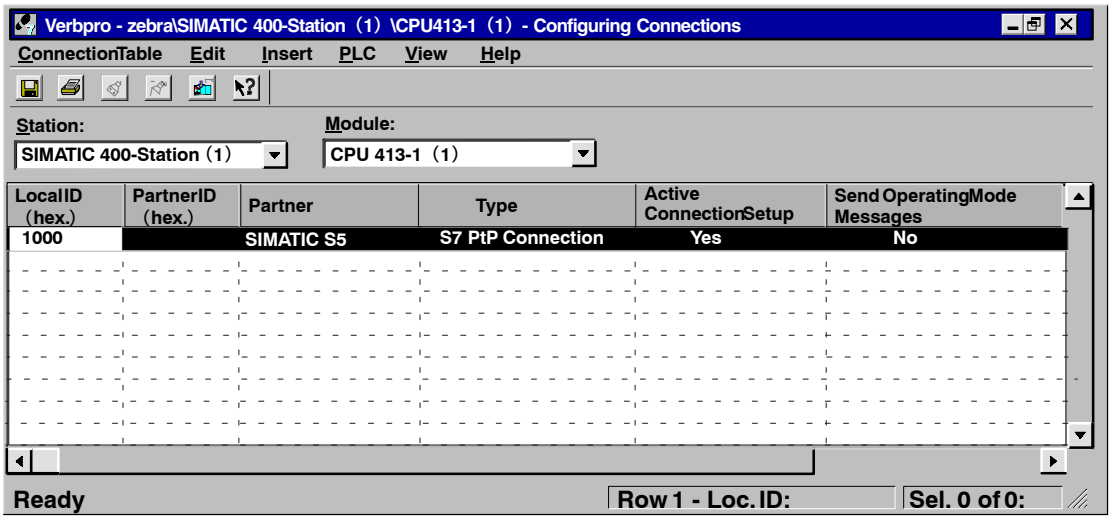

図 3-3　　接続テーブル

3.6 プログラムの挿入および編集

この節で説明する手順は、新規プログラムを作成する場合に適用されます。

3.6.1 ソフトウェア作成の基本手順

概要

CPU 用のソフトウェアはプログラムコンテナに格納されます。SIMATIC S7 モジュールでは、このようなオブジェクトを “S7 Program” と呼びます。

次の図は、SIMATIC 300 ステーションの CPU の S7 プログラムを示しています。

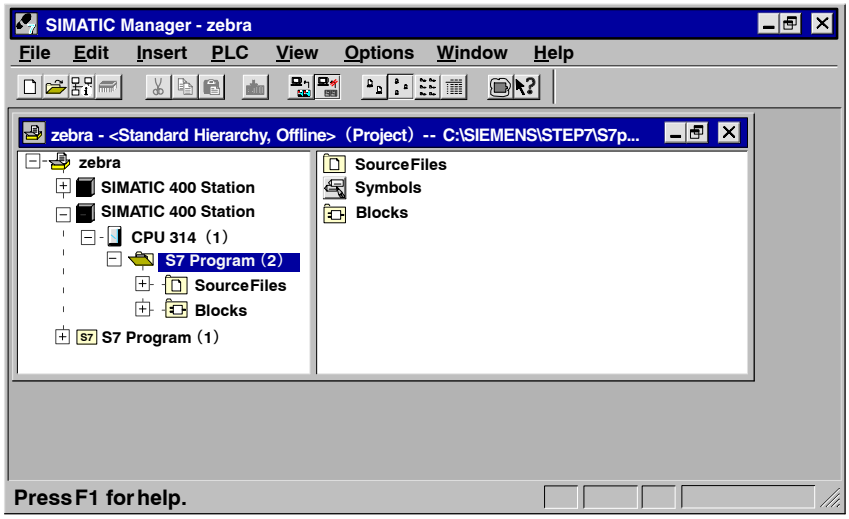

図 3-4　　SIMATIC マネージャで S7 プログラムを開く

手順

プロジェクトのソフトウェアを作成するには、次の手順に従います。

1. S7 プログラムを開きます。
2. S7 プログラムで “Symbols” オブジェクトを開き、シンボルを定義します（このステップは後で実行してもよい）。シンボルの定義の詳細について 第 3 章 13.2 節を参照してください。
3. ブロックを作成する場合は“Blocks” コンテナを開き、ソースファイルを作成する場合は “Source Files” コンテナを開きます。
4. ブロックまたはソースファイルのいずれか該当する方を挿入します（詳細については、第 3 章 6.2 節を参照）。このとき、次のメニューコマンドのいずれかを使用します。
   - **挿入 ▸ S7 ソフトウェア ▸ ブロック**
   - **挿入 ▸ S7 ソフトウェア ▸ ソースファイル**
5. ブロックまたはソースファイルを開き、プログラムを入力します。プログラムの詳細については、「Programming Manuals **/232/-/236/**」を参照してください。

タスクによっては、必ずしも上記の手順すべてを実行する必要はありません。

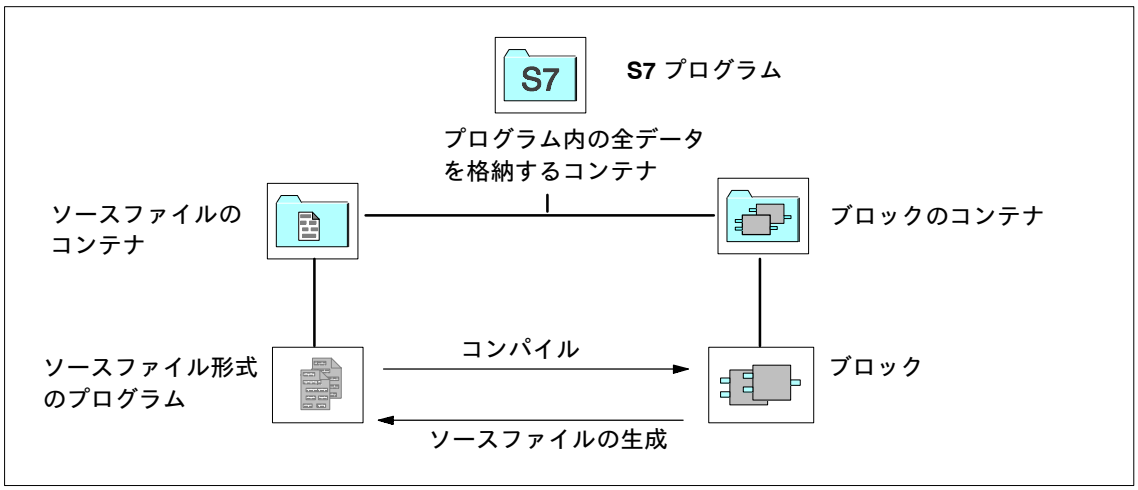

図 3-5　　STEP 7 プロジェクトの基本オブジェクトとその階層構造

3.6.2 ソフトウェアを作成するために S7/M7 プログラムにコンポーネントを挿入

既存のコンポーネント

S7/M7 プログラムは、ソフトウェアのコンテナと同様に、各プログラマブルモジュールに対して自動的に作成されます。

次のオブジェクトは S7 プログラム内ですでに作成されています。

- シンボルテーブル（“Symbols” オブジェクト）
- 先頭ブロックである OB1 ブロックの “Blocks” コンテナ
- ソースファイル形式のプログラム用の “Source Files” コンテナ

S7 ブロックの作成

ステートメントリスト、ファンクションブロックダイアグラム、ラダ-ロジックでプログラムを作成する場合は、既存の “Blocks” オブジェクトを選択し、メニューコマンド [**挿入 ▸ S7 ソフトウェア ▸ ブロック**k] をクリックします。サブメニューで、作成するブロックのタイプ（たとえば、データブロック、ユーザ定義データタイプ（UDT））、ファンクション、ファンクションブロック、オーガニゼーションブロック、変数テーブル（VAT）など）を選択できます。

ここで（空の）ブロックを開き、ステートメントリスト、ラダーロジック、またはファンクションブロックダイアグラムのプログラムの入力を開始できます。詳細については、「*Statement List* **/232/**, *Ladder Logic* **/233/**」および「*Function Block Diagram* **/236/**」の各プログラミングマニュアルを参照してください。

ユーザプログラムにある “System Data” オブジェクト（SDB）はシステムによって作成されたものです。このオブジェクトを開けば内容を表示できますが、一貫性が損なわれるため、内容変更はできません。このオブジェクトを使って、プログラムをロードしたらコンフィグレーションを変更し、変更内容をプログラマブルコントローラにダウンロードします。

標準ライブラリのブロックの使用

ソフトウェアで提供されている標準ライブラリのブロックを使用して、ユーザプログラムを作成することもできます。ライブラリへは、メニューコマンド [**ファイル ▸ 開く**] でアクセスします。標準ライブラリの使用方法とユーザ自身のライブラリを作成する方法については、オンラインヘルプを参照してください。

ソースファイルの作成

ソースファイルをステートメントリストで作成する場合は、S7 プログラムの“Source Files” オブジェクトまたは “Charts” オブジェクトを選択し、さらにメニューコマンド [**挿入 ▸ S7 ソフトウェア ▸ ソースファイル**] を選択します。サブメニューで、プログラミング言語に対応するソースファイルを選択できます。ここで、空のソースファイルを開き、プログラムの入力を開始できます。

シンボルテーブルの作成

S7 プログラムを作成すると、（空の）シンボルテーブル（“Symbols” オブジェクト）が自動的に作成されます。シンボルテーブルを開くと、[シンボルエディタ] ウィンドウが現れ、シンボルテーブルが表示されます。このシンボルテーブルでシンボルを定義できます（詳細については第 3 章 13.2 節を参照）。

外部ソースファイルの挿入

ソースファイルは、ASCII エディタで作成し、編集することができます。作成したソースファイルは、プロジェクトにインポートし、実行ブロックにコンパイルできます。外部ソースファイルを挿入するには、次の手順に従います。

1. ソースファイルのインポート先となる “Source Files” コンテナを選択します。
2. メニューコマンド [**挿入 ▸ 外部ソースファイル**] を選択します。
3. ダイアログボックスが表示されるので、ここにソースファイル名を入力します。

ソースファイルのインポート時に作成したブロックはコンパイルされ、“Blocks” コンテナに保存されます。

3.7 ブロック

3.7.1 比較

次の表に STEP 5 と STEP 7 のブロックを比較します。この表により、STEP 5 ブロックに対応する STEP 7 ブロックがわかります。

絶対的な対応ではない

STEP 7 のブロック環境ではプログラミングオプションが追加されているため、この表で対照されているブロック同士が、常に1対1の対応関係にあるわけではありません。表のブロックは、STEP 7 プログラミングの開始にあたって望ましいブロックであると考えてください。

表 3-2 ブロックの比較: STEP 5 / STEP 7

STEP 5 ブロック	STEP 7 ブロック	説明
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（OB）	オペレーティングシステムのインターフェース
内臓特殊 OB	システムファンクション（SFC） システムファンクションブロック（SFB）	ユーザプログラムで呼び出せる特殊オーガニゼーションブロック（STEP 5）に代わって、STEP7のシステムファンクションが使用されています。
ファンクションブロック（FB, FX）	ファンクション（FC）	STEP 7 ファンクション（FC）は、STEP 5 ファンクションブロックと同じプロパティ
プログラムブロック（PB）	ファンクションブロック（FB）	プログラムブロックは、STEP 7 のファンクションブロックに対応します。STEP 7 ファンクションブロックは、同じ名前の STEP 5のブロックに比べて、プロパティはまったく異なり、新しいプログラミングオプションを提供します。注プログラムブロックは、変換中にファンクション（FC）に転送されます。
シーケンスブロック（SB）	-	STEP 7 にはシーケンスブロックはありません。
データブロック（DB, DX）	データブロック（DB）	STEP 7 と STEP 5 を比べると、データブロックのサイズは STEP 7 の方が大きくなっています（S7-300 では最大 8 Kbyte、S7-400 では最大 64 Kbyte）。
特殊ファンクションのデータブロック DX0、DB1	システムデータブロック（SDB）（CPU パラメータ割り付け）	STEP 7 のシステムデータブロックには、すべてのハードウェアコンフィグレーションデータが入っています。これには、プログラム処理を決定する CPU パラメータ割り付けなどがあります。
コメントブロック DK、DKX、FK、FKX、PK	-	STEP7では、コメントブロックはありません。コメントは、オフラインデータベースの各ブロックに格納されています。

3.7.2 ファンクションおよびファンクションブロック

ファンクション（FC）

ファンクション（FC）は、“メモリ”をもたないロジックブロックです。出力パラメータには、処理されたファンクションの計算値が示されます。FC の呼び出し後に、実パラメータをどのように使用し、保存するのかはユーザが決めます。

ファンクションとファンクションブロックを混同しないでください。
STEP 7 では両者は別のタイプのブロックです。

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、”メモリ”をもつロジックブロックです。メモリはインスタンスデータブロックの形をとっています。インスタンスデータブロックは、ファンクションに関連付けられ、この中に、ファンクションブロックの実パラメータとスタティックデータが保存されています。

ファンクションブロックは、コントローラ構造のプログラミングなどの用途に使用されます。

3.7.3 データブロック

データブロックには、ユーザプログラムのデータが格納されています。以下に示すように、データブロックには、共有データブロックとインスタンスデータブロックがあります。

- 共有データブロックは、特定のブロックには割り付けられません（STEP 5 の場合と同様）。
- インスタンスデータブロックは、ファンクションブロック（FB）に関連付けられ、FB データのほかに、定義されている場合はマルチプルインスタンスのデータが格納されています。

どのデータブロックも、共有データブロックにもインスタンスデータブロックにもできます。

3.7.4 システムブロック

システムファンクション（SFC）とシステムファンクションブロック（SFB）

ファンクションはユーザ自身がプログラミングする必要はありません。たとえば、CPU のオペレーティングシステムで使用できるコンフィグレーション済みブロックを使って、コミュニケーションファンクションをプログラミングすることができます。このファンクションには以下のようなものがあります。

- **システムファンクション（SFC）**：ファンクション（FC）と同じプロパティをもつ
- **システムファンクションブロック（SFB）**：ファンクションブロック（FB）と同じプロパティをもつ

システムデータブロック（SDB）

前の説明は、ユーザプログラムのプログラムまたはデータを含むブロックに関するものでした。これらのブロックのほかに、モジュールパラメータやアドレスなどの設定を含むブロックがあります。これらのブロックを**システムデータブロック（SDB）**と呼び、ハードウェアコンフィグレーションを入力したり、接続テーブルを作成するときに特殊な STEP 7 アプリケーションによって作成されます。

3.7.5 オーガニゼーションブロック

オーガニゼーションブロック（OB）は、オペレーティングシステムとユーザプログラムのインターフェースとなります。各オーガニゼーションブロックはそれぞれ、独自のタスクを実行します。

オーガニゼーションブロックの配布

S7 CPU の STL ユーザプログラムは、オートメーションソリューションに必要なオーガニゼーションブロック（OB）から組み立てます。

表 3-3 S5 と S7 の OB の比較

機能		S5	S7
メインプログラム	フリーサイクル	OB1	OB1
割り込み	時間遅延割り込み	OB6	OB20 ～ OB23
	時刻（クロック制御）割り込み	OB9	OB10 ～ OB17
	ハードウェア割り込み	OB2 ～ OB5	OB40 ～ OB47
	プロセス割り込み	OB2 ～ OB9（IB 0）	ハードウェア割り込みに変更
	周期的（時間指定）割り込み	OB10 ～ OB18	OB30 ～ OB38
	マルチコンピューティング割り込み	-	OB60
スタートアップ	手動完全（コールド）再起動	OB21（S5-115U） OB20（S5-135Uより）	OB100
	手動（ウォーム）再起動	OB21（S5-135Uより）	OB101
	自動（ウォーム）再起動	OB22	OB101
エラー	エラー	OB19 ～ OB35	OB121, OB122, OB80 ～ OB87
その他	STOP モードの処理	OB39	省略
	バックグラウンド処理	-	OB90

エラー処理

エラー OB

エラー OB は、プログラムの実行中にエラーが発生すると呼び出されます。エラー OB を使用して、プログラムエラーの処理を行うことができます。特定のエラータイプにエラー OB がない場合は、CPU は STOP モードになります。

表 3-4　S5 と S7 でのエラー処理

機能	S5	S7
ロードされていないブロックの呼び出し	OB19	OB121
I/O モジュールへの直接アクセスによりタイムアウト	OB23	OB122
プロセスイメージと IPC フラグ（プロセッサ間通信フラグ）の更新でタイムアウト	OB24	OB122
アドレス指定エラー	OB25	OB122
周期時間の超過	OB26	OB80
置換エラー	OB27	省略
オペレータによる停止	OB28 (S5-135U)	省略
入力バイト IB 0 によりタイムアウト	OB28 (S5-155U)	OB85
命令コードが不正	OB29 (S5-135U)	STOP
拡張アドレス領域の I/O への直接アクセスによりタイムアウト	OB29 (S5-155U)	OB122
パラメータが不正	OB30 (S5-135U)	省略
パリティエラー、またはユーザメモリへのアクセスでタイムアウト	OB30 (S5-155U)	OB122
特殊ファンクショングループエラー	OB31	省略
データブロックによるロード/転送エラー	OB32	OB121
時間指定割り込みの衝突	OB33	OB80
コントローラエラー	OB34 (S5-135U)	省略
データブロック生成エラー	OB34 (S5-155U)	SFC フィードバック
通信エラー	OB35	OB84

S5 と S7 のトラブルシューティング

超過した信号

S5 では、ステータスワードビット OV と OS を使って、超過した信号の報告を評価することができます。S5 と S7 の動作ではそれほどの違いはありません。

命令を参照するステータスビットの動作については、「*Statement List Programming Manual* **/232/**」を参照してください。

内蔵特殊ファンクション

S5 CPU におけるユーザプログラムとシステムプログラムのインターフェースは、オペレーティングシステム領域へのアクセスによって、または特殊 OB を介して形成されます。

S7 CPU では、このインターフェースには、オーガニゼーションブロックのほかに、2 つの新しいブロックタイプ（"システムファンクション"と"システムファンクションブロック"）があります。

システムファンクション/システムファンクションブロック

システムファンクション（SFC）とシステムファンクションブロック（SFB）は、CPU オペレーティングシステムに組み込まれているブロックで、必要に応じて STEP 7 ユーザプログラムで呼び出すことができます。システムファンクション（SFC）の処理中にエラーが発生すると、リターン値 RET_VAL を使ってユーザプログラムでこのエラーを評価することができます。

表 3-5　　S5 と S7 の特殊ファンクション

機能	S5 ブロック	S7 のブロック
周期的タイムトリガ	OB31	SFC43 RE_TRIGR
バッテリ異常	OB34	OB81（エラー処理はユーザによる設定が可能）
条件コードバイトへのアクセス	OB110	STEP 7 命令: L STW/T STW
ACCU 1 - 4 の削除	OB111	STEP 7 命令シーケンス: L 0; PUSH; PUSH; PUSH
ACCU のロールアップ	OB112	同等機能なし: STEP 7 命令: PUSH
ACCU のロールダウン	OB113	同等機能なし STEP 7 命令: POP
すべての割り込みのオン/オフを無効にする	OB120	SFC41 DIS_AIRT SFC42 EN_AIRT
周期（時間指定）割り込みの個々のオン/オフを無効にする	OB121	SFC39 DIS_IRT SFC40 EN_IRT
すべての割り込みのオン/オフを遅延させる	OB122	SFC41 DIS_AIRT SFC42 EN_AIRT
周期（時間指定）割り込みの個々のオン/オフを遅延させる	OB123	SFC39 DIS_IRT SFC40 EN_IRT
CPU 時間の設定/読み取り （次ページに続く）	OB150	SFC0 SET_CLK SFC1 READ_CLK

表 3-5　　S5 と S7 の特殊ファンクション、続き

機能	S5 ブロック	S7 のブロック
時間制御割り込み時間の設定/読み取り	OB151	SFC28 SET_TINT SFC30 ACT_TINT SFC31 QRY_TINT
周期的統計値	OB152	OB1 のローカルデータ
カウンタループ	OB160 - 163 (S5-135U)	STEP 7 命令: LOOP
可変時間指定ループ	OB160 (S5-115U)	SFC47 WAIT
ブロックスタックの読み取り	OB170	省略
可変データブロックアクセス	OB180	省略
データブロックのテスト	OB181	SFC24 TEST_DB
データ領域のコピー	OB182	SFC20 BLKMOV
フラグをデータブロックへ転送	OB190, OB192	SFC20 BLKMOV
データブロックをフラグ領域へ転送	OB191, OB193	SFC20 BLKMOV
マルチプロセッサ通信の機能	OB200 - 205	省略
ページアクセス	OB216 - 218	S7 ではページアドレス指定なし
符号拡張子	OB220	S7 命令: ITD
周期的モニタ時間の設定	OB221	S7 によるパラメータ割り付け
周期的モニタ時間の再開始	OB222	SFC43 RE_TRIGR
スタートアップタイプの比較	OB223	同一スタートアップタイプにのみマルチコンピューティングを開始
ブロックで IPC フラグを転送	OB224	省略
システムプログラムからのワードの読み取り	OB226	省略
システムプログラムからの CRC の読み取り	OB227	省略
プログラム処理レベルのステータス情報の読み取り	OB228	SFC51 RDSYSST SFC6 RD_SINFO
ハンドリングブロックのファンクション	OB230 - 237	SFB による通信
シフトレジスタの初期化	OB240	省略
シフトレジスタの処理	OB241	省略
シフトレジスタの削除	OB242	省略
コントロール: PID アルゴリズムの初期化 コントロール: PID アルゴリズムの処理	OB250 OB251	クローズドループコントロール FB: FB41 - FB43 または SFB41 - SFB43
データブロック（DB/DX）の DB RAM への転送	OB254, OB255	省略

3.7.6 変換中のブロック表示

ブロック割り付け

S7 ではブロック構造が変更されています。次の図は、STEP 5 と STEP 7 における変換プロセス後のブロック割り付け例を簡単に示しています。

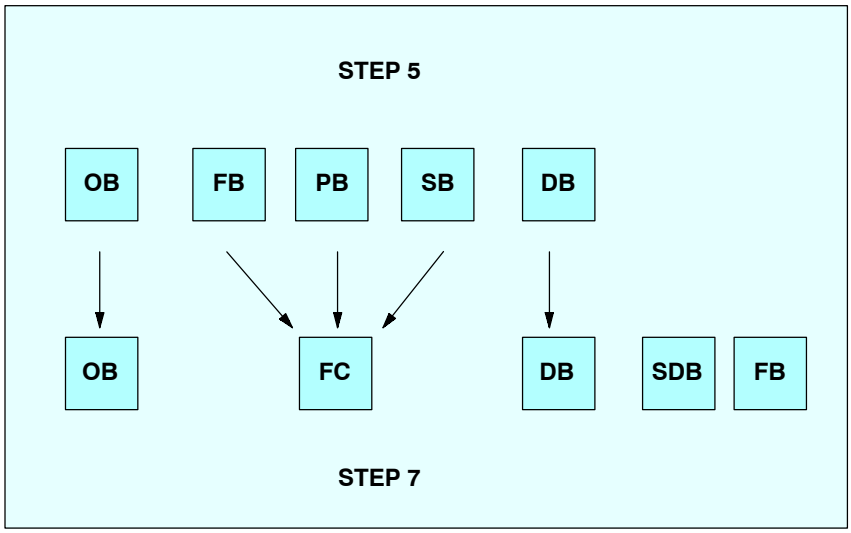

図 3-6 STEP 5 と STEP 7 で同等機能をもつブロック

次ページ の表 3-6 では、ブロック呼び出しの変換方法を示しています。

表 3-6　　S5 と S7 のブロックタイプ

S5			S7	
OB	固定番号	ユーザプログラム	対応する S7 OB	固定番号
OB	固定番号	特殊ファンクション	変換できない、S7 による再プログラムが必要	
PB	0 ～ 255	ユーザプログラム	パラメータなしの FC ブロック	番号が明示される
FB/FX	0 ～ 255	ユーザプログラム	名前が保持されているパラメータをもつ FC ブロック	番号が明示される
FB	固定番号	内蔵 ファンクションブロック	FBlib1 ライブラリ内のロード可能 FC で、コンパイルする前に、変換されたされたファイルにロードする必要がある	固定番号
FB/FX	固定名称	標準ファンクションブロック	FBlib1 ライブラリ内のロード可能 FC で、コンパイルする前に、変換されたされたファイルにロードする必要がある	固定番号
SB	0 ～ 255	ユーザプログラム	パラメータなしの FC ブロック （シーケンサは変換できないので、S7GRAPHで作成する必要がある）	番号が提示される
DB	2 ～ 255	ユーザプログラム	共有データブロック（DB）	S5 の番号
DX	1 ～ 255	ユーザプログラム	共有データブロック（DB）	256 未満の番号が提示される
DB1/ DX0		システム設定付きのデータブロック	ブロックに CPU 固有のエントリがある場合、パラメータ設定は STEP 7 で行わなければならない。DB1 と DX0 の変換された内容は関連性がないので、削除することができる。	

3.8 システム設定

DB1 と DX0 の変換　次の表に、DB1 と DX0（システム設定）においてパートナーの機能がどのように作成されるのかを示します。

表 3-7　DB1 からのシステム設定の変換

S5 パラメータブロック	S7 での実装方法
スタートアップ遅延	SFC47 WAIT を呼び出す
IPC フラグ	グローバルデータ通信を使って設定し、以下を呼び出す SFC60 GD_SND SFC61 GD_RCV
エラーコードの場所	システムにより、診断バッファにエラーメッセージが入力される。”エラーコードの場所”に関する情報は省略されている。
組み込み FB の名前を置換する	省略
オンボードアナログ入力	CPU プロパティを使って HWConfig で設定
オンボード割り込み	CPU プロパティを使って HWConfig で設定
オンボードカウンタ	CPU プロパティを使って HWConfig で設定
OB の優先度の変更	CPU プロパティを使って HWConfig で設定
プロセスイメージの出力/無効化	SFC27 UPDAT_PO を呼び出す
プロセスイメージの読み込み/無効化	SFC26 UPDAT_PI を呼び出す
保持フラグ	CPU プロパティを使って HWConfig で設定
保持タイマ	CPU プロパティを使って HWConfig で設定
保持カウンタ	CPU プロパティを使って HWConfig で設定
SINEC L1	MPI バス（グローバルデータ通信）に変更
SINEC L2	HWConfig により設定
ソフトウェア保護	準備中
クロックパラメータ	CPU プロパティを使って、または SFC28 SET_TINT 呼び出して HWConfig で設定
パラメータの時間指定 OB への割り付け	CPU プロパティを使って HWConfig で設定
周期的時間モニタ	CPU プロパティを使って HWConfig で設定

表 3-8　　DX0 からのシステム設定の変換

S5 パラメータブロック	S7 での実装方法
アドレス指定エラーのモニタ	OB121 を呼び出す
IPC フラグの更新	グローバルデータ通信
電源投入後のスタートアップ	CPU プロパティを使って HWConfig で設定
マルチプロセッサオペレーションで同期化を開始	CPU プロパティを使って HWConfig で設定
タイマセルの番号	CPU 固有の固定値（S7-300）、または CPU プロパティを使って HWConfig で設定（S7-400）
エラー処理	呼び出し: SFC36 MSK_FLT SFC37 DMSK_FLT
浮動小数点演算	あり
プロセス（ハードウェア）割り込みトリガ	CPU プロパティを使って HWConfig で設定
時間指定（周期）割り込み処理モード	SFC28 SET_TINT を呼び出す
周期的時間モニタ	CPU プロパティを使って HWConfig で設定

3.9 標準ファンクション

変換の実行中、S5 の標準ファンクションは、同じ機能をもつ変換済みファンクションに自動的に置き換わります。S7 では、これらのファンクションのほとんどは、簡単なコマンドシーケンスで置き換えることができます。これにより、メモリは維持され、サイクルタイムは短縮されます。

標準ファンクションは、プログラムコンテナ FBLib1 内の “StdLib30” S7 にあります。

ライブラリの使用方法については、オンラインヘルプを参照してください。

3.9.1 浮動小数点演算

STEP 5	STEP 7		STEP 5	STEP 7	
FB 名	番号	名称	FB 名	番号	名称
GP:FPGP	FC61	GP_FPGP	GP:MUL	FC65	GP_MUL
GP:GPFP	FC62	GP_GPFP	GP:DIV	FC66	GP_DIV
GP:ADD	FC63	GP_ADD	GP:VGL	FC67	GP_VGL
GP:SUB	FC64	GP_SUB	RAD:GP	FC68	RAD_GP

3.9.2 シグナルファンクション

STEP 5	STEP 7		STEP 5	STEP 7	
FB 名	番号	名称	FB 名	番号	名称
MLD:TG	FC69	MLD_TG	MLD:EZ	FC75	MLD_EZ
MELD:TGZ	FC70	MELD_TGZ	MLD:ED	FC76	MLD_ED
MLD:EZW	FC71	MLD_EZW	MLD:EZWK	FC77	MLD_EZWK
MLD:EDW	FC72	MLD_EDW	MLD:EDWK	FC78	MLD_EDWK
MLD:SAMW	FC73	MLD_SAMW	MLD:EZK	FC79	MLD_EZK
MLD:SAM	FC74	MLD_SAM	MLD:EDK	FC80	MLD_EDK

3.9.3 内蔵ファンクション

STEP 5	STEP 7	
FB 名	番号	名称
COD:B4	FC81	COD_B4
COD:16	FC82	COD_16
MUL:16	FC83	MUL_16
DIV:16	FC84	DIV_16

3.9.4 基本ファンクション

STEP 5	STEP 7		STEP 5	STEP 7	
FB 名	番号	名称	FB 名	番号	名称
ADD:32	FC85	ADD_32	REG:LIFO	FC93	REG_LIFO
SUB:32	FC86	SUB_32	DB:COPY	FC94	DB_COPY
MUL:32	FC87	MUL_32	DB:COPY	FC95	DB_COPY
DIV:32	FC88	DIV_32	RETTEN	FC96	RETTEN
RAD:16	FC89	RAD_16	LADEN	FC97	LADEN
REG:SCHB	FC90	REG_SCHB	COD:B8	FC98	COD_B8
REG:SCHW	FC91	REG_SCHW	COD:32	FC99	COD_32
REG:FIFO	FC92	REG_FIFO			

3.9.5 アナログファンクション

STEP 5	STEP 7		STEP 5	STEP 7	
FB 名	番号	名称	FB 名	番号	名称
AE:460	FC100	AE_460_1	AE:466	FC106	AE_466_1
AE:460	FC101	AE_460_2	AE:466	FC107	AE_466_2
AE:463	FC102	AE_463_1	RLG:AA	FC108	RLG_AA1
AE:463	FC103	AE_463_2	RLG:AA	FC109	RLG_AA2
AE:464	FC104	AE_464_1	PER:ET	FC110	PER_ET1
AE:464	FC105	AE_464_2	PER:ET	FC111	PER_ET2

3.9.6 数値演算ファンクション

STEP 5	STEP 7		STEP 5	STEP 7	
FB 名	番号	名称	FB 名	番号	名称
SINE	FC112	SINE	ARCCOT	FC119	ARCCOT
COSINE	FC113	COSINE	LN X	FC120	LN_X
TANGENT	FC114	TANGENT	LG X	FC121	LG_X
COTANG	FC115	COTANG	B LOG X	FC122	B_LOG_X
ARCSIN	FC116	ARCSIN	E^X	FC123	E_H_N
ARCCOS	FC117	ARCCOS	ZEHN^X	FC124	ZEHN_H_N
ARCTAN	FC118	ARCTAN	A2^A1	FC125	A2_H_A1

3.10 データタイプ

STEP 7 では、新しいデータフォーマットが使用されています。次の表に、S5 と S7 のデータタイプを比較します。

表 3-9　　S5 と S7 のデータタイプ

S5 のデータタイプ	S7 のデータタイプ	データクラス
BOOL, BYTE, WORD, DWORD, 整数, 倍長整数, 実数, タイム値, - ASCII 文字	BOOL, BYTE, WORD, DWORD, INT, DINT, REAL, S5TIME, TIME, DATE; TIME_OF_DAY, CHAR	基本データタイプ
-	DATE_AND_TIME, STRING, ARRAY, STRUCT	複合データタイプ
タイマ, カウンタ, ブロック - -	TIMER, COUNTER, BLOCK_FC, BLOCK_FB, BLOCK_DB, BLOCK_SDB, POINTER, ANY	パラメータタイプ

表 3-10　　S5 と S7 における整数のフォーマット

S5 のフォーマット	**例**	**S7 のフォーマット**	**例**
KB	L KB 10	k8	L B#16# A
KF	L KF 10	k16	L 10
KH	L KH FFFF	16#	L 16# FFFF
KM	L KM 1111111111111111	2#	L 2# 11111111_11111111
KY	L KY 10,12	B#	L B#（10,12）
KT	L KT 10.0	S5TIME#（S5T#）	L S5TIME# 100ms
KC	L KC 30	C#	L C#30
DH	L DH FFFF FFFF	16#	L DW#16# FFFF_FFFF
KS	L KS WW	’ xx ’	L ’ WW ’
KG	L KG +234 +09	実数	L +2.34 E+08
表現: S5 フォーマット ← 指数 →　← 仮数 → 31 30　24　23　22　0 SE 2^6.. ... ... 2^0　SM 2^{-1}..... .....2^{-23} 指数 = 指数の値 SE = 指数の符号 SM = 仮数の符号 値の範囲: 1.5 x 10^{-39} to 1.7 x 10^{38}		**表現: ANSI/IEEE に適合するシングルフォーマット** ← 指数 →　← 仮数 → 31 30　23　22　0 S 2^7.. ... ... 2^0　2^{-1}.. .... 2^{-23} 指数 = 実指数 + バイアス*（+127） S = 仮数の符号 値の範囲: 約 1.18 x 10^{-38} ～ 3.4 x 10^{+38}	

* バイアス: これは、指数を正数部と負数部に分けるオフセット係数。指数部の値 127 は、絶対値の 0 に相当します。

データタイプの詳細については、「*Statement List Programming Manual* /**232**/」を参照してください。

3.11 アドレス領域

3.11.1 概要

表 3-11　　S5 と S7 のアドレス

アドレス領域	S5 のアドレス	S7 のアドレス	備考
入力	I	I	
出力	Q	Q	
I/O	P, Q, G	ロードコマンドにPI	共有 I/O なし
		転送コマンドにPQ	変換済み
ビットメモリ（フラグ）領域	F	M	
	S	M	M 256.0 より（コンバータ）
	“スクラッチパッドフラグ”	L	フラグと同様に変換
タイマ	T	T	
カウンタ	C	C	
データ領域	D...	DB...	共有データアドレスとして変換
システムデータ	RS, RT, RI, RJ	-	変換しない
ページ領域	C	-	変換済み

データアドレスに関する注記

S7 では、DB レジスタと DI レジスタという 2 つのデータブロックレジスタがあります。DB レジスタは主に共有データブロックに使用され、DI レジスタはインスタンス DB に使用されます。このため、データアドレスにも 2つのタイプがあります。アドレス DBX、DBB、DBW、DBD は共有データブロックのアドレスで、アドレス DIX、DIB、DIW、DID はインスタンス DB のアドレスです。変換中は、共有データブロックのアドレスはデータブロックアドレス D、DB、DW、DD に使用されます。

データブロックの変換方法も確認してください（第3章7.6節を参照）。

危険

アドレス領域のサイズと番号の領域および S7 ブロックの番号と長さの領域はすべて、使用する CPU に依存しています。CPU 性能の基準およびレベルについては、第2章2.1節を参照してください。

3.11.2 S7 の新しいアドレス: ローカルデータ

STEP 7 のローカルデータ

STEP 7 のローカルデータは、ロジックブロックに割り付けられたデータで、宣言セクションまたは変数宣言で宣言されます。ブロックに応じて、仮パラメータ、スタティックデータ、テンポラリデータで構成されます。ローカルデータは常にシンボルでアドレス指定されます。

ブロックパラメータ

ファンクション（FC）のブロックパラメータは、S5 のブロックパラメータと同様に扱われます。ブロックパラメータは、対応する実パラメータへのポインタです。

ファンクションブロック（FB）のブロックパラメータは、スタティックローカルデータと同様にインスタンスブロックに保存されます。

スタティックローカルデータ

スタティックローカルデータは、すべてのファンクションブロックで使用できます。このローカルデータは、宣言セクションで定義され、インスタンスデータブロックに保存されます。

スタティックローカルデータは、共有データブロックのデータアドレスと同様に、プログラムによって上書きされるまでその値が保持されます。

一般的に、スタティックローカルデータは、ファンクションブロックでのみ処理されます。ただし、このローカルデータはデータブロックに保存されるため、ユーザプログラムからいつでもアクセスすることができます。これは、共有データブロックの変数の場合と同じです。

テンポラリローカルデータ

STEP 5 のスクラッチパッドフラグ

STEP 5 では、ブロック内でのデータの一時保存場所としてビットメモリアドレス領域が使用されます。共通の取り決めにより、フラグ 200 ～ 250 は一時保存場所として確保されています。スクラッチパッドフラグの管理は、ユーザに完全に任されています。

STEP 7 のテンポラリローカルデータ

テンポラリローカルデータはデータ保存領域で、ブロック処理の間だけ有効です。ブロックの処理が終了すると、これらのローカルデータは直ちに使用していたメモリを解放します。優先度クラスごとに専用のローカルデータスタックが設けられています。これにより、割り込みプログラムによって中間結果が誤って上書きされるのを防ぎます。

STEP 7 でのテンポラリローカルデータの使用

STEP 7 では、次の3種類の用途に使用されます。

- ユーザプログラムのデータの中間保存場所

  この用途については前述したとおりで、ファンクション（FC）、ファンションブロック（FB）、オーガニゼーションブロック（OB）に適用されます。

- オペレーティングシステムの情報をユーザプログラムに転送するために使用するメモリ

  オペレーティングシステムがユーザプログラムに提供する情報を「スタート情報」と呼びます。スタート情報は、オペレーティングシステムとユーザプログラム間のインターフェースとして、オーガニゼーションブロック（OB）にのみ提供されています。

- FC でパラメータを転送する

テンポラリローカルデータの宣言場所

テンポラリローカルデータはブロック内で宣言します。ブロックを新規に作成したら、テンポラリ変数のシンボル名を宣言し、それからブロック内で使用します。S7-300 では、優先度ごとに 256 バイトを使用できます。S7-400 では合計16 Kbyte を使用でき、CPU にパラメータを割り付けるときに優先度ごとに分割することができます。

3.12 命令

次の表に、使用する命令の概要を示します。さらに、変換可能な命令も示します。変換できない命令については、その他の変換オプションを示しています。

表 3-12　　S5 および S7 の命令

命令のタイプ	S5	S7	変換	変換オプション
アキュムレータ命令	TAK, ENT, I, D, ADDBN, ADDKF, ADDDH	TAK, ENT, INC, DEC, +, S7の新規命令: CAW, CAD, PUSH, POP, LEAVE	可	-
アドレスレジスタ命令 / レジスタ命令	MA1, MBR, ABR, MAS, MAB, MSB, MSA, MBA, MBS; TSG, LRB, LRW, LRD, TRB, TRW, TRD	S7の新規命令: LAR1, LAR2, TAR1, TAR2, +AR1, +AR2, CAR	不可	アドレスレジスタ（AR1，AR2）を使用する
ビット命令	A, AN, O, ON, A（, O（, ）, O, S, R, RB, RD= TB, TBN, SU, RU	A, AN, O, ON, A（, O（, ）, O, S, R, = SET; A, SET; AN, SET; S, SET; R New in S7: X, XN, X（, XN（, FP, FN, NOT, SET, CLR, SAVE	可	-
タイマ命令	SP, SE, SD, SS/SSU, SF/SFD, FR, SEC	SP, SE, SD, SS, SF, FR, S T	可	-
カウンタ命令	CU/SSU, CD/SFD, FR, SEC	CU, CD, FR, S C	可	-
ロード命令と転送命令	L, LD, LW, LDW, TL PB, L QB, L PW, L QW, T PB, T QB, T PW, T QW	L, LC, T L PIB, L PIW, T PQB, T PQW	可	-
（次ページに続く）	LY GB / GW / GD / CB / CW / CD, LW GW / GD / CW / CD, TY GB / GW / GD / CB / CW / CD, TW GW / GD / CW / CD		不可	I/O 領域へアクセスする

表 3-12　　S5 および S7 の命令、続き

命令のタイプ	S5	S7	変換	変換オプション
整数演算命令	+F, -F, xF, :F, +D, -D	+I, -I, *I, /I, +D, -D, *D, /D S7の新規命令: MOD	可	-
浮動少数点演算命令	+G, -G, xG, :G	+R, -R, *R, /R	可	-
比較命令	!=F, ><F, >F, <F, >=F, <=F, !=D, ><D, D, <D, >=D, <=D, !=G, ><G, >G, <G, >=G, <=G	==I, <>I, >I, <I; >=I, <=I, ==D, <>D, >D, <D, >=D, <=D, ==R, <>R, >R, <R, >=R, <=R	可	-
変換命令	CFW, CSW, CSD DEF, DED, DUF, DUD, GFD, FDG	INVI, NEGI, NEGD, BTI, BTD, DTB, ITB, RND, DTR S7の新規命令: ITD, RND+, RND-, TRUNC, INVD, NEGR	可	-
ワードロジック命令	AW, OW, XOW	AW, OW, XOW S7の新規命令: AD, OD, XOD	可	-
シフト命令と回転命令	SLW, SLD, SRW, SRD, SVW, SVD, RLD, RRD	SLW, SLD, SRW, SRD, SSI, SSD, RLD, RRD S7の新規命令: RLDA, RRDA	可	-
データブロック命令	G, CX	OPN	可	
	G, GX	SFC22	不可	SFC22 CREATE_DB を呼び出す
（次ページに続く）		S7の新規命令: CDB L DBLG, L DBNO, L DILG, L DINO		

表 3-12　　S5 および S7 の命令、続き

命令のタイプ	S5	S7	変換	変換オプション
論理制御命令、ジャンプ	JU, JC, JN, JZ, JP, JM, JO, JOS, JUR	JU, JC, JN, JZ, JP, JM, JO, JOS S7の新規命令: JCN, JCB, JNB, JBI, JNBI, JMZ, JPZ, JUO, LOOP, JL	可	-
ブロック命令	JU, JC, DOU, DOC, BE, BEU, BEC	CALL, BE, BEU, BEC	可	-
コマンド出力命令/マスタコントロールリレー命令	BAS, BAF	S7の新規命令: MCRA, MCRD, MCR（, ）MCR	不可	SFC26、SFC27、またはマスタコントロールリレー命令を呼び出す
停止コマンド	STP, STS, STW	SFC46	不可	SFC46 STP を呼び出す
プロセシングファンクション	DO <仮パラメータ>	-	不可	DB/コードの呼び出しを新規にプログラムする必要がある
	DO FW, DO DW	メモリ間接アドレス指定	可	推奨: レジスタ間接アドレス指定を行う
	DO RS	領域間レジスタ間接アドレス指定	不可	間接アドレス指定で代用する必要がある（第3章13.4節を参照）
絶対メモリアドレス指定	LIR, TIR, LDI, TDI	-	不可	間接アドレス指定で代用する必要がある（第3章13.4節を参照）
ブロック転送	TNB, TNW, TXB, TXW	SFC20	不可	SFC20 BLKMOV を呼び出す
割り込みコマンド	LIM, SIM, IAE, RAE, IA, RA	SFC39 ～ 42	不可	SFC39 - 42 を呼び出す

（次ページに続く）

表 3-12　　S5 および S7 の命令、続き

命令のタイプ	S5	S7	変換	変換オプション
ページコマンド	ACR, TSC, TSG	-	不可	S7 ではページアクセスはない
数値演算ファンクション	-	ABS, COS, SIN, TAN, ACOS, ASIN, ATAN, EXP, LN	-	-
ヌル命令	BLD xxx NOP 0, NOP 1	BLD xxx NOP 0, NOP 1	可	-

3.13 アドレス指定

3.13.1 絶対アドレス指定

S5 と S7 では絶対アドレス指定の方法は同じですが、1つだけ相違点があります。

S7 では、データブロックのデータは**バイト単位**でアドレス指定されるため、S5 のワードアドレスには 2 を掛けてバイトアドレスに変換します。

次の表に、変換実行中の割り付けを示します（データ領域アドレス指定）。

S5	S7
DL 0, 1, 2, 3, ...255	DBB 0, 2, 4, 6, ...510
DR 0, 1, 2, 3, ...255	DBB 1, 3, 5, 7, ...511
DW 0, 1, 2, 3, ...255	DBW 0, 2, 4, 6, ...510
DD 0, 1, 2, 3, ...254	DBD 0, 2, 4, 6, ...508
D x.y	8 ≤ y ≤ 15ではDBX 2 x.y 0 ≤ y ≤ 7ではDBX（2 x+1）.y

3.13.2 シンボルによるアドレス指定

S5 でのシンボルによるアドレス指定は S7 でも使用されます。ただし、シンボルの作成および使用に関するオプションが新たに追加されています。プログラミング方法は従来のものと同じです。

STEP 5のシンボル

STEP 5 プログラムのシンボルは、シンボルエディタを使って宣言します。このエディタによって割り付けリストが作成され、絶対アドレスの代わりにこのエディタで定義されたシンボルを使用することができます。

STEP 7のシンボル

S7 では、シンボルに24文字まで使用できます。

共有シンボル

STEP 7 にもシンボルエディタがありますが、割り付けリスト（ZULI）は「シンボルテーブル」と呼ばれています。シンボルテーブルでは、入力、出力、ビットメモリ（フラグ）、ブロックなどのすべての共有シンボルを宣言できます。

シンボルエディタでシンボルを割り付けると、これらのシンボルは CPU プログラムで有効になります。

ローカルシンボル

STEP 7 では、シンボルテーブルでシンボルを宣言できるほかに、ブロックのプログラミング時にデータアドレスとローカルデータ領域のローカルシンボルを指定するためのオプションが用意されています。

シンボルエディタを使用せずに、ブロック内でシンボルの割り付けを行う場合、割り付けたシンボルは接続されているブロックでのみ「有効」になります。この場合、シンボルは「ブロックに対してローカル」になります。

シンボルを宣言する場所

STEP 7 では、シンボルを指定する時期は正確には決まっていません。次のいずれかを選択できます。

- プログラミングを開始する前に指定する
  （ユーザプログラムが徐々に入力される場合、つまり各行が作成されてからプログラム構文がチェックされる場合は、この方法にする必要がある）
- ユーザプログラムを作成してからコンパイルするまでの間に指定する
  （ユーザプログラムがフリーエディットモードで入力される場合、つまりプログラムを ASCII ファイル（ソースファイル）で作成する場合は、この方法にする必要がある）

シンボルテーブルのインポート

S7 では、好きなエディタを使ってシンボルテーブルの作成および編集を行うことができます。

他のツールで作成したテーブルをシンボルテーブルにインポートし、編集することができます。たとえば、インポート機能を使えば、STEP 5/ST で作成した割り付けリストを変換し、それから追加することができます。

使用できるデータフォーマットは、*.SDF、*.ASC、*.DIF、*.SEQ です。

シンボルテーブルをインポートするには、次の手順に従います。

1. シンボルテーブルが含まれている S7 プログラムをプロジェクトウィンドウで開きます。
2. “Symbols” コンテナをダブルクリックして、シンボルテーブルを開きます。
3. シンボルテーブルが表示されているウィンドウでメニューコマンド [**シンボルテーブル ▸ インポート**] を選択します。ダイアログボックスが表示されます。
4. インポートするシンボルテーブルをダイアログボックスで選択し、[開く] ボタンをクリックします。
5. シンボルテーブルのデータレコードを調べて、必要であれば修正を行います。
6. シンボルテーブルを保存して閉じます。

注

S5 から S7 へ変換された *.SEQ ファイルフォーマットのシンボルテーブルは、S5 にインポートすることはできません。S5 と S7 の間でシンボルテーブルを変換する場合は、ファイルフォーマット *.DIF を推奨します。

シンボルテーブルの詳細については、「*User Manual* **/231/**」を参照してください。

3.13.3 新機能: データアドレスの完全指定

完全アドレス指定では、データブロックをデータアドレスと一緒に指定します。この方法は S5 では使用できません。

このアドレス指定は、絶対方式またはシンボル方式で実行することができます。1つのステートメントで絶対アドレス指定とシンボルアドレス指定を混在させることはできません。

例

L DB100.DBW6

L DB_MOTOR.SPEED

DB_MOTOR は、データブロック DB100 のシンボルで、シンボルテーブルで定義されます。MOTOR.SPEED は、データブロックで宣言されたデータアドレスです。つまり、データアドレスのシンボル名（DB_MOTOR.SPEED）は絶対アドレス（DB100.DBW6）と同様に一意になります。

完全アドレス指定によるデータアクセスは、共有データブロックレジスタ（DB レジスタ）に関してのみ実行できます。完全アドレス指定の実行中、STL エディタはステートメントを発行します。

1. DB レジスタ（たとえば OPN DB100）レジスタを介してデータブロックを開く
2. データアドレス（たとえば L DBW 6）へアクセスする

完全アドレス指定のデータアクセスによるオペレーション

指定されているデータアドレスのデータタイプに対応するすべての命令に対して、完全アドレス指定によるアクセスが可能です。

完全アドレス指定されたデータアドレスは、ブロックパラメータとしても指定できます。この方法は、データブロックが呼び出されるとその時点で切り替えることができるため、強く推奨します。完全アドレス指定により、適正なデータブロックから適正なデータアドレスが転送されます。

「部分アドレス指定」の危険性

原則として、データアドレスへのアクセスは STEP 5 と同じ方法で実行できます（「部分アドレス指定」）。

例:

```
L DBW 6
L SPEED
```

STEP 7 でデータアドレスへのアクセスを行うと、様々なオペレーションの実行中に S7-300/S7-400 CPU のレジスタが変更されるため、問題が発生する場合があります。DB レジスタの DB 番号が上書きされる場合もあります。

DB レジスタは、以下の場合に上書きされる可能性があります。したがって、特に注意が必要です。

- 完全アドレス指定によるデータアクセスの実行中に DB レジスタが上書きされる
- ファンクションブロック（FB）が呼び出されると、呼び出し側ブロックのデータブロックレジスタが上書きされる
- パラメータを転送するファンクション（FC）が複合データタイプ（たとえばSTRING、DATE_AND_TIME、ARRAY、STRUCT、UDT）で呼び出されると、呼び出し側ブロックの DB レジスタの内容が上書きされる
- DB（たとえばDB100.DBX0.1）に保存されている FC に実パラメータを割り付けると、STEP 7 では、DB（DB100）が DB レジスタの内容を上書きされた状態で開く
- FBが複合データタイプ（たとえばSTRING、DATE_AND_TIME、ARRAY、STRUCT、UDT）で入出力パラメータをアドレス指定すると、STEP 7 は DB レジスタを使ってデータにアクセスする。このステップで、DB レジスタの内容は上書きされる
- FCが複合データタイプ（たとえばSTRING、DATE_AND_TIME、ARRAY、STRUCT、UDT）でパラメータ（入力、出力 入出力）をアドレス指定すると、STEP 7 は DB レジスタを使ってデータにアクセスする。このステップで、DB レジスタの内容は上書きされる

3.13.4 間接アドレス指定

S5 の “DO” ファンクションによる間接アドレス指定は、S7 では新規の間接メモリおよびレジスタアドレス指定コマンドに置き替えられています。

STEP 5 のポインタフォーマット

S5では、示されたプロセシングオペレーションのポインタに1ワードが使用されます。このポインタの構造を図 3-7 に示します。

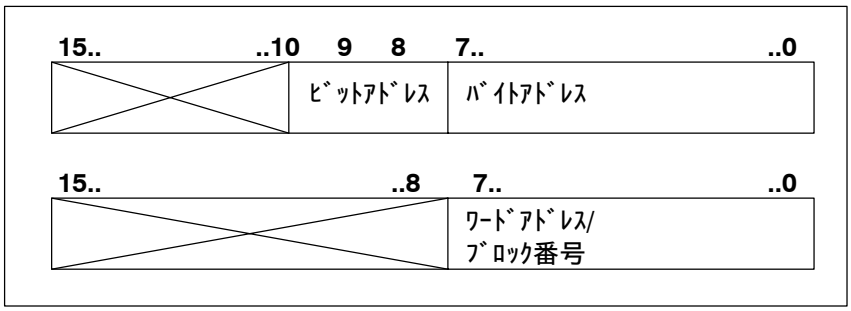

図 3-7　　ポインタ S5 の構造

STEP 7 のポインタフォーマット

S7 では、2つのポインタフォーマット（ワードおよびダブルワード）が使用されます。

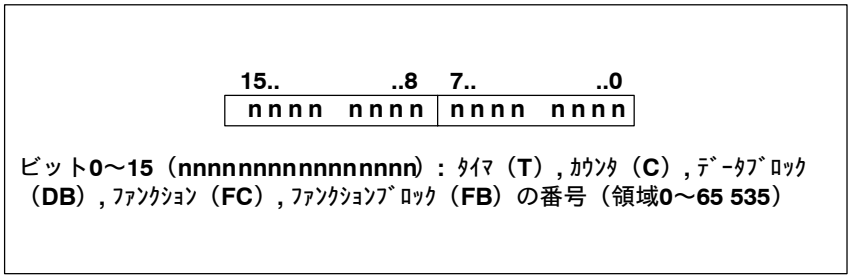

図 3-8　　メモリ間接アドレス指定のポインタ（ワードフォーマット）

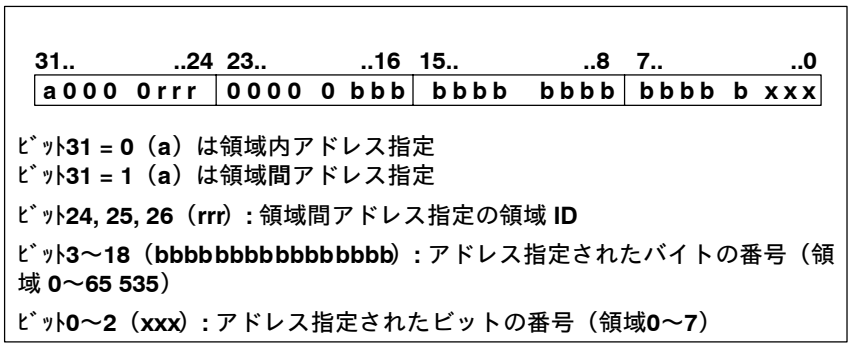

図 3-9　　メモリ間接アドレス指定とレジスタ間接アドレス指定のポインタ（ダブルワードフォーマット）

メモリ間接アドレス指定

メモリ間接アドレス指定は、S5 の間接アドレス指定に相当します。メモリ間接アドレス指定の実行中、このアドレスはこの値のアドレスを示し、命令の処理を行います。アドレスは次のパートで構成されます。

- アドレス識別子。たとえば、”入力バイト” の “IB”
- タイマ（T）、カウンタ（C）、データブロック（DB）、ファンクション（FC）、またはファンクションブロック（FB）の番号で構成されるワード
- アドレス識別子で示されるメモリ領域内の値の正確なアドレスを指定するダブルワード

このアドレスでは、値や番号のアドレスを間接的に指定するポインタが使用されます。このワードまたはダブルワードは、次の領域のいずれかに配置することができます。

- ビットメモリ（フラグ） （M）
- データブロック （DB）
- インスタンスデータブロック （DI）
- ローカルデータ （L）

メモリ間接アドレス指定の利点は、プログラムの編集時にステートメントのアドレスをダイナミックに変更できることです。

例

次の例は、ワードフォーマットのポインタを使用する方法を示しています。

STL S5	STL S7	説明

`L KB 5`
`T FW 2`
`DO FW 2`
`L T 0`
	`L +5`
	`T MW 2`
	
	`L T [MW 2]`
		値5を整数として ACCU 1 にロードする
		ACCU 1 の内容をメモリワード MW2 に転送する
		タイマ T 5 のタイマ値をロードする

次の2つの例は、ダブルワードフォーマットのポインタを使用する方法を示しています。

STL S5	STL S7	説明

`L KB 8`
`T FY 3`
`L KB 7`
`T FY 2`
`DO FW 2`
`A I 0.0`
`DO FW 2`
`= Q 0.0`
	`L P#8.7`
	`T MD 2`
	
	`A I [MD 2]`
	
	`= Q [MD 2]`
		2#0000 0000 0000 0000 0000 0000 0100 0111（2進数）を ACCU 1（S7）にロードする
		アドレス 8.7 をメモリワード FW 2（S5）/ メモリダブルワード MD 2（S7）に保存する
		
		コントローラが入力 I 8.7 の問合せを行い、その信号状態を出力 Q 8.7 に割り付ける

STL S5	STL S7	説明
L KB 8 DO FW 2 DO FW 2 L IB 0 DO FW 2 T FW 0	L P#8.0 T MD2 L IB [MD2] T MW [MD2]	2#0000 0000 0000 0000 0000 0000 0100 0000（2進数）をACCU 1（S7）にロードする アドレス 8 をメモリワード FW 2（S5）/ メモリダブルワード MD 2（S7）に保存する コントローラが入力バイト IB 8 をロードし、and メモリワード FW 8（STEP 7 では MW 8 in STEP 7）の内容を転送する

正確なシーケンス構文の使用

データブロックのメモリ領域に保存されているメモリ間接アドレスを使用する場合、[データブロックを開く] 命令（OPN）を使用してまず最初にデータブロックを開く必要があります。この後、データワードまたはデータダブルワードを間接アドレスとして使用することができます（次の例を参照）。

```
OPN     DB10
L       IB [DBD 20]
```

バイト、ワード、またはダブルワードにアクセスする場合、ポインタのビット番号が “0” であることを確認します。

レジスタ間接アドレス指定

STEP 7 では、アドレスレジスタ AR1 と AR2 は間接アドレス指定に使用されます。

間接アドレス指定では、アドレスは値のメモリロケーションを指定し、命令を処理します。アドレスは次の2つのパートで構成されます。

- アドレス識別子
- アドレスレジスタの内容に追加されるオフセットを示すアドレスレジスタおよびポインタ。これは、命令が処理するアドレスを正確に決定するために使用される。ポインタは、P#Byte.Bitで示される。

アドレスは、値のアドレスを間接的にポイントします。この場合、アドレスレジスタとオフセットが使用されます。

領域内のレジスタ間接アドレス指定を使用する命令では、アドレスレジスタの値は変更されません。

詳細については、「*Statement List Programming Manual* **/232/**」を参照してください。

Teil 1: Planung des Umstiegs 第2部: プログラムの変換

4 手順

STL の場合、S7 と S5 のプログラミングはほとんど同じです。同様に ラダーのプログラミングも S7 と S5では同じで、S7 でのFBD のプログラミングも S5 CSF と同じです。このように、S5 ユーザが既存のプログラムを S7 で使用する場合、変更ははるかに簡単です。試用を行い、テストした S5 プログラムをもとに新しいシステムを構築し、S7 プログラムに変換します。

作業の進め方

次のリストでは、S5 プログラムを変換する方法を説明し、必要な情報が記載されているセクションをリストします。

このリストは、一例なので、ガイドラインとして使用してください（各ステップをスキップすることもできます）。

4.1 S5 システムの分析

S5 プログラムを変換する前に、次の項目を明確にする必要があります。

モジュールの機能（第2章を参照）

S5 プログラムで使用しているモジュールの機能はどのような方法で S7 で実現できるのか？ アダプタまたはインターフェースモジュールを使用すれば、S5 モジュールを S7 で使用できるのか？ S5 モジュールを S7 モジュールと置き替えることができるのか？

システム設定（第3章8節を参照）

必要なシステム設定を S7 でインプリメントできるか？

命令の範囲（第3章12章を参照）

S5 CPU で使用する命令範囲をS7 CPU でどのようにインプリメントできるのか？

各命令を変換できない場合、対応するプログラムセクションを示すメッセージが出力され、すべての命令を手動でプログラミングし直さなければなりません。

標準ソフトウェア（第3章9節を参照）

変換するプログラムで呼び出される S5 標準ファンクションブロックが、S7 にファンクションとして存在しているか？

付属の S7 標準ソフトウェアには、基本ファンクション、浮動小数点演算ファンクション、組み込みファンクション、シグナルファンクション、数値演算ファンクション用に変換された標準ソフトウェアパッケージが含まれています。

特殊ファンクション（表 3-5 を参照）

S5 プログラムで使用する組み込み特殊ファンクションを置き替えることができるか？

プログラムのどのパートをS7で再プログラムするのか？

一般に、プログラムのすべてのパートを変換できるわけではありません。次の項目を参考にすれば、S5 プログラムを変換すべきなのか、S7 で作成し直すべきなのかを決定できます。

- デジタル論理演算と2進論理演算のみを含むプログラムは変換する必要はない
- 絶対アドレスは、S7　　ではアドレス指定できない。対応する命令（LIR、TIR など）は変換されない。絶対アドレスで問題が頻繁に発生する場合、プログラムの該当パートや、必要ならばプログラム全体を書き直すとよい
- プロセシングファンクション（DO FW や DO DW など）は部分的に変換される。ただし、この場合、S7 でこのファンクションを再プログラミングすることにより、メモリを節約できる。この機能は、間接アドレス指定によって実現できる。
- 変換の実行中、実パラメータは変換されずに転送されるので、ブロック呼び出しのパラメータ値を常にチェックし、調整する必要がある

4.2 S7 プロジェクトの作成

STEP 7 では、次の2種類の方法でプロジェクトを作成できます。

STEP 7ウィザードによりプロジェクトを作成する

STEP 7 ウィザードにより、使用する CPU で STEP 7 プログラムを素早く作成することができます。このステップが完了したら、プログラミングを開始できます。

プログラムを手動で作成する

プロジェクトは、手動でも作成できます。作成手順については、第3章3.1節で説明しています。

4.3 ハードウェアの定義

HWConfig でデータが決定したら、このデータを使って変換準備を行えるので、この時点でハードウェアをコンフィグレーションするとよいでしょう。

ただし、ハードウェアをまだコンフィグレーションしたくない場合は、後で行ってもかまいません。

ハードウェアの定義

第2章（「ハードウェア」）の情報は、ユーザのコンフィグレーションに必要な S7 モジュールまたは S5 モジュールを選択し、ハードウェアコンフィグレーションテーブルに入力を行うのに役立ちます（第3章4節を参照）。

アドレスの割り付け

モジュールへのアドレスの割り付けは、HWConfig によって自動的に実行されます。つまり、変換中にアドレスを使用することができます。

システム設定の実行

HWConfig で CPU にパラメータを割り付ける場合、DB1/DX0 で作成したシステム設定を S5 で実行するか、あるいはシステムユーティリティで実行します（第3章4節を参照）。

保持動作の指定

保持動作は、CPU のパラメータデータで設定することもできます。ただし、この動作は、バッテリバックアップに依存します（第3章4節を参照）。

変換の準備

5

概要

必要なファイルを用意する（第5章1節を参照）	• プログラムファイル <Name>ST.S5D • クロスリファレンスリスト <Name>XR.INI • オプションの割り付けリスト <Name>Z0.SEQ
アドレスをチェックする（第5章2節を参照）	• アドレス番号 • ブロック番号
S5 プログラムを準備する（第5章3節を参照）	• データブロックDB1 / DX0 の評価と削除 • 組み込みブロックからの呼び出しの削除 • システムデータ領域へのアクセスの削除 • アドレス領域の調整 • 変換できないプログラムパートへのマクロの割り付け • ストラクチャをもたないデータブロックは1データワードになるまで削除する
マクロを作成する（第5章4節を参照）	• コマンドマクロ • オーガニゼーションブロック（OB）マクロ

5.1 必要なファイルを用意する

S5 プログラムを変換するための基本データとして、次のデータが必要です。

- プログラムファイル <Name> ST.S5D
- クロスリファレンスリスト <Name> XR.INI

クロスリファレンスリストは、S5 プログラムのプログラム構造と呼び出し階層を保持するために変換時に必要です。

その他の条件

プログラムで絶対アドレスではなくシンボルアドレスを使用する場合は、変換済みの割り付けリストを作成するために次のデータが必要です。

- S5 割り付けリスト <Name> Z0.SEQ.

準備手順

変換の準備を行うには、次の手順に従います。

1. S5 ソフトウェアを使って、S5 プログラム用の現在のクロスリファレンスリストを作成します。
2. STEP 5 プログラムファイル、対応するクロスリファレンスリスト、さらに必要ならば割り付けリストを DOS ディレクトリにコピーします。

5.2 アドレスのチェック

CPUのファンクションの範囲

変換したプログラムは、使用する S7 CPU に合わせて調整しなければならない場合があります。

S7 CPU のファンクションの範囲に関する概要を得るには、次の手順に従います。

1. 使用する S7 CPU を決定します。
2. この S7 CPU については第2節2.1節の性能仕様の表を参照し、次の2つの仕様を比較してください。

- アドレス番号
- ブロック番号

アドレスとブロックを使用する場合

1. SIMATIC マネージャを開きます。
2. プロジェクト構造のオンラインビューで S7 CPU を選択します。
3. メニューコマンド [**PLC ▸ モジュール情報**] を使って、次の情報を提供するダイアログボックスを開きます。

- [全般] タブで、CPU タイプを明確にし、メモリ構成の情報を取得し、アドレス領域の使用可能サイズを読みます。
- [ブロック] タブに、使用可能なブロックの情報が表示されています。これには、各ブロックタイプの最大数と最大長、CPU にあるすべての OB、SFB、SFC の最大数および最大長が含まれています。

変換したプログラムの調整

変換している STL プログラムを CPU で実行できるように調整するには、ブロックおよびアドレスの許容数を確認し、必要に応じて修正します。

5.3 S5 プログラムの準備

STEP 5 プログラムを実際に変換する前に、STEP 7 プログラムとして使用するための準備を行えます（ただし、必ずしも最初に準備を行う必要はありません。必要な修正は、変換後に STEP 7 ソースファイルで行うこともできます）。最初に調整を行えば、変換時に発生するエラーメッセージや警告を減らすことができます。

たとえば、変換を実行する前に、STEP 5 プログラムに対して次の調整を行えます。

- プログラムプロパティ DB1 または DX0 を使って、データブロックのシステム設定を評価することができます。DB1 と DX0 は、評価後に削除できます。
- 内臓ブロックからすべての呼び出しを削除するか、システムデータ領域へアクセスします。この機能は、S7 CPU へパラメータを割り付けることによりアクティブにできます。
- STEP 5 の機能「再配線」を使って、すべての入力、出力、ペリフェラルアドレス領域を（新規の）モジュールアドレスに合わせて調整します（調整を実行する際、STEP 5 アドレス領域の範囲を超えないようにします。この範囲を超えると、変換プロセスの最初のサイクルでエラーが報告されます。エラーが発生すると、これらの命令は変換されません）。
- 繰り返し実行される変換不能なプログラムパートは、各プログラムパートごとに STEP 5 命令を「1つだけ」残して削除します。この命令にマクロに割り付けて、プログラムパートと置換することができます（第5章4節を参照）。
- ストラクチャをもたない（長い）データブロック（データバッファとして使用されるデータブロックなど）が多数含まれている場合、これらのデータブロックのデータワードを1つだけ残して、すべて削除することができます。変換してからコンパイルするまでの間に、ソースファイル内のこれらのデータブロックの内容をアレイ宣言（ARRAY [1..256] of WORD）を使って、バッファなどとしてプログラミングすることができます。

コンバータを使えば、プログラム全体のほかに、プログラムブロックごとに変換することもできます。

5.4 マクロの作成

マクロの使用

変換時には、以下に対してマクロを定義できます。

- 自動的に変換できない S5 命令
- 標準変換以外の方法で変換したい S5 命令

上記のような S5 命令が多数含まれているプログラムでは、マクロは有益です。

マクロファンクション

マクロは以下を取り替えることができます。

- S5 命令
- S5 オーガニゼーションブロック（OB）

マクロは、S7S5CAPA.MAC ファイルの SIMATIC 命令セットおよび S7S5CAPB.MAC ファイルの国際命令セットに対して保存されます。両方の命令セットを使用する場合、各ファイルごとにマクロを指定する必要があります。マクロには命令マクロと OB マクロがあります。どちらのマクロもそれぞれ 256 個まで作成できます。

5.4.1 命令マクロ

命令マクロは次のように構成します。

$MACRO: <S5 命令**>**

S7 命令シーケンス

$ENDMACRO

マクロを定義する場合、<S5 命令> に対して完全なステートメント（命令および絶対アドレス）を入力します。

次の表に、S5 でデータブロックのセットアップに使用されるステートメント G DB 0 のマクロを示します。セットアップするデータブロックの長さ（ワード単位）は ACCU 1 に格納されています。S7 では、この機能は、システムファンクション SFC22 CREAT_DB を使って実行します。データブロックの長さはバイト単位で変換しなければなりません

表 5-1 マクロ命令の例

マクロ	S5	S7

$MACRO: G DB 0 //Replaces instruction
//for setting up a DB
 	
SLW 1 //Number of words
//into number of bytes
T MW 102
CALL SFC 22(//Call SFC CREAT_DB
LOW_LIMIT := MW 100,
UP_LIMIT := MW 100,
COUNT := MW 102,
RET_VAL := MW 106,
DB_NUMBER := MW 104);
$ENDMACRO
	L Constant
	DO FW 100
	
	G DB 0
		L Constant;
		
		SLW 1;
		T MW 102;
		CALL SFC22(
		LOW_LIMIT := MW 100,
		UP_LIMIT := MW 100,
		COUNT := MW 102,
		RET_VAL := MW 106,
		DB_NUMBER := MW 104);

5.4.2 OB マクロ

S5 と S7 のオーガニゼーションブロックには相違があるため、S5 OB を使って命令の変換を制御するとよいでしょう。この場合、OB マクロは次のように構成しなければなりません。

$OBCALL: <OB の番号**>**

CALL <S7 システムファンクション**>;**

$ENDMACRO

S5 ソースファイルにアドレス OB x の命令がある場合、この命令を定義済みのマクロ命令に取り替えることができます。ただし、OB を仮パラメータとして使用する FB 呼び出しは例外です。

表 5-2 OB マクロの例

マクロ	S5	S7
$OBCALL: 31 //Replaces instructions **//with OB31** **CALL SFC 43;** **$ENDMACRO**	**JU OB 31**	**CALL SFC43;**

OB 作成の注記

S5 と S7 のオーガニゼーションブロックは、その機能に違いがあります。自動的に変換できない OB は以下によって置き替える必要ができます。

- ファンクションの範囲が異なる OB
- 新規の S7 命令
- ハードウェアパラメータの割り付け時に定義されたシステム設定

S5 OB の置き替えの詳細については、ダイアログボックス3章7.5節を参照してください。

注

マクロが2回定義されたかどうかのチェックは行われません。定義が2回行われると、最初に定義したマクロが使用されます。また、指定された S7 命令シーケンスが正しいかどうかのチェックも行われません。キーワードおよび特殊文字（コロン）は正しく記述するようにします。

5.4.3 マクロの編集

マクロは次の方法で作成します。

1. Windows 95 のタスクバーの [スタート] ボタンをクリックし、[Simatic/STEP 7/Convert S5 files] を選択して、S5/S7 コンバータを開始します。
2. メニューコマンド [**編集 ▸ マクロの置換**] を選択します（プログラムファイルは閉じているはずです!）。

結果: S7S5CAPB.MAC ファイルが開きます。

3. マクロを前述のように入力し、メニューコマンド [**ファイル ▸ 保存**] で保存します。
4. メニューコマンド [**ファイル ▸ 閉じる**] でファイルを閉じます。

結果: S7S5CAPB.MAC ファイルが閉じます。このマクロは、次の変換が開始するまで有効です。

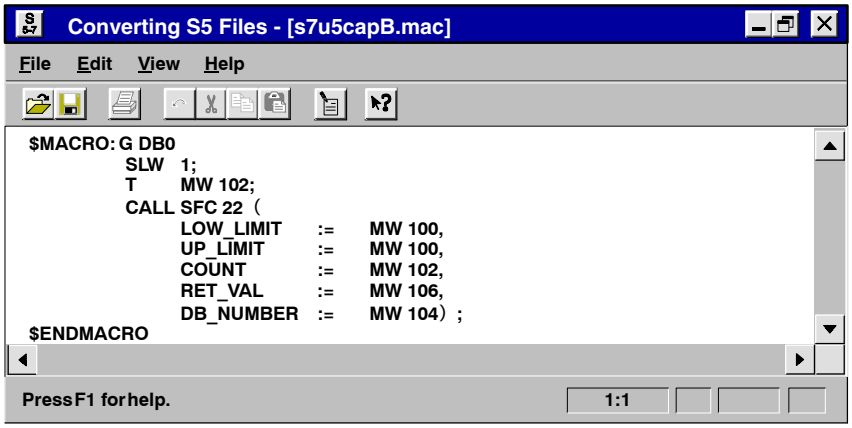

図 5-1 [S5 ファイルの変換] ウィンドウのマクロ

6 変換

6.1 変換の開始

前提条件

プログラムの変換を開始する前に、変換対象の S5 ファイル、クロスリファレンスリスト、およびオプションの割り付けリストが同じディレクトリ内に置かれているかどうかを確認します（5.1 節を参照）。

S5/S7 コンバータ

プログラミング装置に STEP 7 ソフトウェアをインストールした後で、Windows 95 のタスクバーの [スタート] ボタンを使用して S5/S7 コンバータを起動します。

- エントリ [Simatic/STEP 7 V2/Converting S5 Files] をクリックします

S5/S7 コンバータは、次の初期画面を表示します。

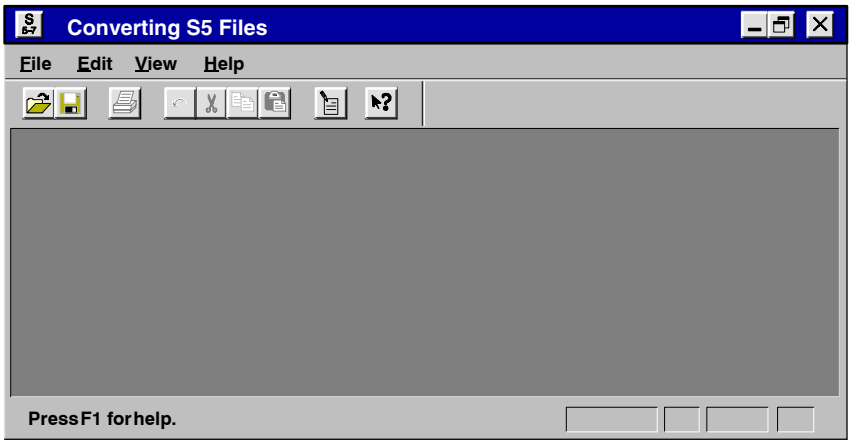

図 6-1　　S5/S7 コンバータの初期画面

プログラムファイルの選択

プログラムファイルを選択するには、以下の手順に従います。

1. メニューコマンド [**ファイル▸　開く**] を選択します。
2. 変換対象のファイルがあるドライブとディレクトリを選択します。
3. 変換対象のファイルを選択してから、確認して [OK] ボタンをクリックします。

結果: S5/S7 コンバータは、ソースファイルと目的ファイル、さらに古いブロック番号と新ブロック番号の割り付けを表示します。

次の図は、ダイアログボックス [S5 ファイルの変換 [<Test>ST.S5D]] を示しています。

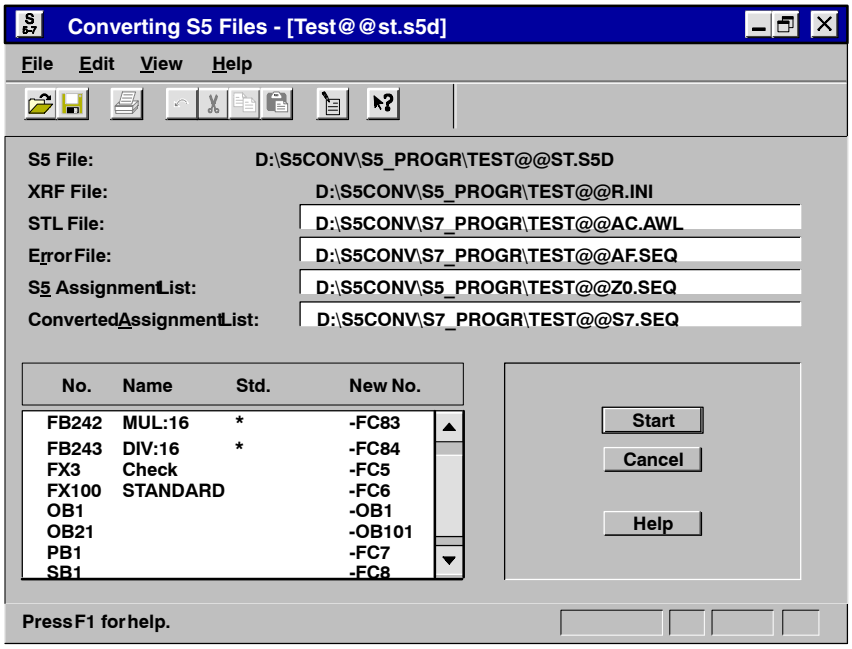

図 6-2　　[S5 ファイルの変換-[<Name>ST.S5D]] ダイアログボックス

目的ファイルの名前の変更

ソフトウェアが提供する目的ファイル “STL File”、“Error File”、および “Converted Assignment List” の名前は、必要に応じて変更することができます。たとえば、エディタを使用して変換されたファイルを処理する場合、そのエディタが特定の命名規則（TEST.TXT など）に基づくファイルしか受け付けなければ、ファイル名の変更が必要となります。

1. 変更したい目的ファイルのパス名が表示されているテキスト領域をクリックします。
2. 必要に応じてテキストを変更します。

割り付け No. -> New No

コンバータは、変換対象のブロックに対して新番号を提供し、それを “Convert S5 File [<Test>ST.S5D]” ダイアログボックスに表示します。提供された番号とは違う番号を割り付けたい場合は、以下の手順に従います。

1. 変更するブロック番号をダブルクリックします。
2. “New Block Number” ダイアログボックスに新しい番号を入力し、確認して “OK” ボタンをクリックします。

S5 標準ファンクションブロック

S5 プログラムに標準ファンクションブロックが含まれている場合は、対応する “Std” 列にアスタリスク（*）が表示されます。

変換の開始

変換を開始するには、“Start” ボタンをクリックします。変換は、２回の変換実行と、割り付けリストの変換から構成されています。

１番目の変換実行では、S5 プログラムが S5 ソースファイルに変換され、すべてのブロックとコメントがそのファイルに取り込まれます。

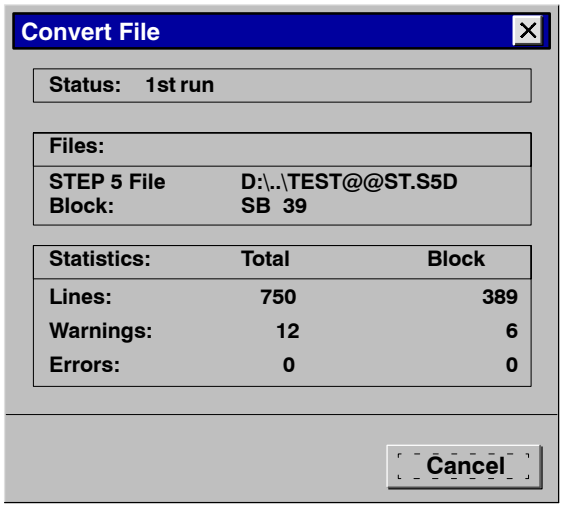

図 6-3　　１番目の変換実行

２番目の変換実行ではS5 ソースファイルが、新しいブロックタイプ、ブロック番号、S7 構文をもつ STL ソースファイルに変換されます。

割り付けリストの変換

S5割り付けリストのシンボルは、シンボルエディタでインポート可能なフォームに変換されます。

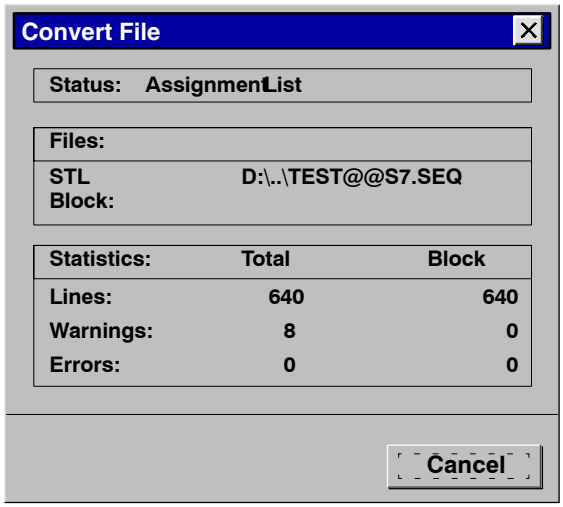

図 6-4　　割り付けリストの変換

6.2 生成されるファイル

変換時には、S5/S7 コンバータにより次のファイルが生成されます。

- ファイル <Name>A0.SEQ:

  このファイルは、最初の変換時に生成されます。これには、ファイル <Name>ST.S5D が ASCII 形式で格納されています。

- ファイル <Name>AC.AWL:

  このファイルは、2番目の変換時に生成されます。これには STL プログラムが格納されています。マクロ定義に誤りがあると、この変換時にエラーメッセージが生成されます。

- ファイル <Name>S7.SEQ:

  このファイルは、割り付けリストの変換により生成されます。このファイルには、シンボルエディタでインポートするのに適したフォームに変換された割り付けリストが格納されています。

- エラーファイル <Name>AF.SEQ:

  このファイルは、[S5 ファイルの変換] ウィンドウの上部リストボックスに表示され、変換されたプログラムのエラーおよび警告が含まれています。これらのメッセージは、変換実行中と割り付けリスト変換中に生成されます。

変換が完了すると、変換したプログラムで生成されたエラーと警告の総数が表示されます。

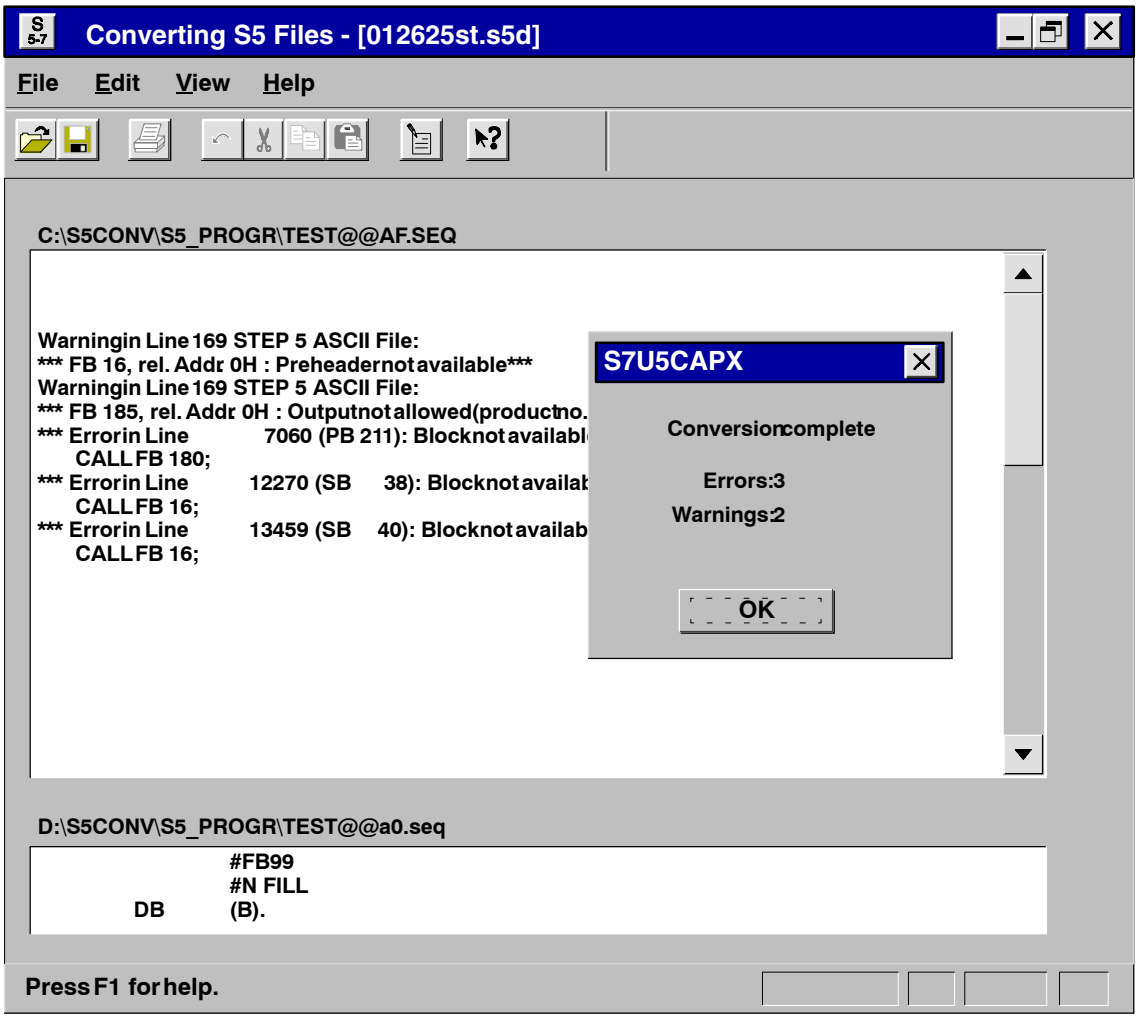

図 6-5　　変換中のメッセージ

エラー発生箇所の表示

ウィンドウの下部リストボックスには、エラーが発生したファイルの場所を表示できます。

プログラム内のエラー検出箇所を示すメッセージが STL ソースファイルに出力されます。このファイルには、問題発生（たとえば、命令内容の変更などが原因）の可能性を示す警告や表示も含まれています。

メッセージの印刷

メニューコマンド[**ファイル ▸ 印刷**] を選択して、必要なメッセージファイルを指定します。

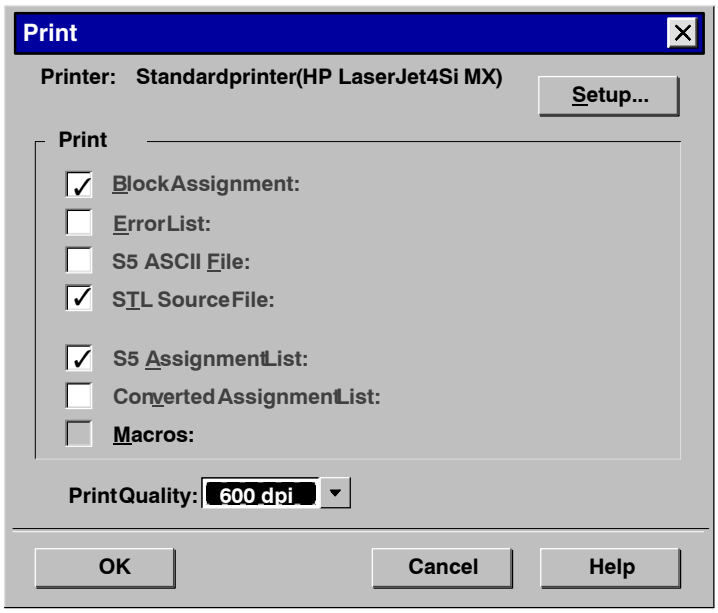

図 6-6　「印刷」ダイアログボックス

6.3 メッセージの解釈

メッセージの分析

メッセージを分析するには、以下の手順に従います。

1. リストボックスに、エラーが格納されたファイルを表示させます。
2. メッセージの意味は、オンラインヘルプで調べることができます。
3. 下表に示す救済策に従ってエラーを訂正します。

エラーメッセージ

S5 プログラムの一部が変換されずに、ただのコメントとして S7 プログラムに取り込まれた場合に、エラーメッセージが表示されます。表 6.1 に、すべてのエラーメッセージと意味、有効な対策を示します。

参照する規則

S5 プログラムから S7 プログラムへの変換時に適用される規則については、第 3 章（ソフトウェア）を参照してください。そこでは、推測されるエラー原因や STL プログラムの後編集に関する情報についても示します。

表 6-1　エラーメッセージの意味と対策

エラーメッセージ	ソース	意味	対策
Absolute parameter dows not match address	1 番目	アドレス ID が間違っています。	命令をチェックします。
Bit access to T/C is no longer allowed (please check)	2 番目	S5 プログラムにタイマおよびカウンタへのビットアクセスが含まれています。	STL プログラムをチェック。
Block not available	1 番目	呼出し対象のブロック（FB、FX）が見つかりません。またはブロックリストに表示されているブロックがプログラムファイル内に存在していません。	プログラム構造をチェック。
	2 番目	プログラムファイル内に存在しないブロックが呼び出されました。	クロスリファレンスリストが指定されてたかどうかをチェック、またはプログラム構造をチェックします。
CALL OB is not allowed	2 番目	OB の呼び出しが S7 では 許可されていません。	必要に応じて、CALL SFC コマンドを使用します。
CALL SFC xy generating, please extend parameter list	2 番目	SFC のパラメータが見つかりません。	SFC パラメータリストを完全なものにします。
Command in block not allowed	1 番目	たとえば、プログラムブロック内のジャンプ。	命令をチェックします。
Comment too long	1 番目	S5 ファイルでエラーが発生。	プログラムファイルチェック。
Conversion error	2 番目	BI に定数が付いていません。	ロード命令を使用し定数指定。
Directory not available	1 番目	プログラムファイルにブロックが存在しません。	プログラムファイルをチェックします。
Error in macro file. Macro xy ignored	2 番目	マクロエラー	マクロ命令をチェックします。
Error in parameter	1 番目	S5 プログラム内でのエラー	プログラムファイルチェック。

表 6-1　　エラーメッセージの意味と対策, 続き

エラーメッセージ	ソース	意味	対策
File not found	全般	選択ファイルが存在しません。	プログラムファイルチェック。
Invalid MC5 code was converted	1 番目	旧式の S5 命令の変換。	なし。
Invalid operator	1 番目	S5 ファイル内の演算子が不明、または変換できません。	その演算子を適切な S7 命令に置き換えてください。
Invalid operator, may be replaced by the instruction \"L P# formal parameter\"	2 番目	このフォームの演算子を S7 にロードすることはできません。	指定された命令を使用する必要があります。
Jump label cannot be generated	2 番目	JUR 命令が、ブロック制限を越えています。	S5 プログラム内の誤りを訂正します。
Label invalid	1 番目	ジャンプラベルに無効な文字が含まれています。	S5 ファイルをチェックします。
Label undefined	1 番目	プリヘッダ内にジャンプラベルが定義されていません。	S5 ファイルをチェックします。
Memory overflow in programming device (space problem)	1 番目	メインメモリの容量が不足しています。	メインメモリ内で不要となったファイルを削除します。
No access rights	全般	ファイルが読み取り専用です。	読み取り専用属性を削除します。
No block name given	1 番目	ブロック名が空白です。	ブロック名を入力します。
Undefined command	1 番目	無効な MC5/STL 命令です。	S5 プログラムファイル訂正。
	2 番目	S7 に該当する命令が存在しません。	マクロを編集するか、問題の命令を適切な S7 命令シーケンスと置き換えます。
Undefined formal parameter	1 番目	ブロック呼び出し時に余計なパラメータが検出されました。	S5 プログラムファイルをチェックします。
Write error on diskette	全般	ファイルが読み取り専用か、ディスケット上に空き領域がありません。	読み取り専用属性を削除するか、不要なデータを削除します。
Wrong address	1 番目	アドレスが命令と不一致。	S5 ソースをチェックします。
	2 番目	アドレスが命令と不一致。	STL ファイルを修正します。
Wrong comment length	1 番目	S5 ファイルのエラー	プログラムファイルチェック。
Wrong nesting depth	1 番目	（）で括られた式の終わりが正しくありません。	ネストレベルをチェックし、プログラム作成上の誤りを訂正。
Wrong number of parameters	1 番目	S5 プログラムでエラーが発生しました。	プログラムファイルをチェックします。
Wrong parameter type	1 番目	S5 プログラムでエラーが発生しました。	プログラムファイルをチェックします。

警告

S5 プログラムの一部が変換されても再度チェックが必要な箇所が存在する場合は、警告が表示されます。

表 6-2　警告の意味と対策

警告	ソース	意味	対策
Jump instruction after EDIT cannot be compiled	2 番目	JU 付きの EDIT を自動的に変換できません。	STLファイル内の該当命令をJUに置き換えジャンプをチェック
ID only influences Accu 1-L, now whole Accu 1	2 番目	S7 アキュムレータは 32 ビットに拡張されています。	STL プログラム内の間接命令 INCREMENT/DECREMENT をチェックします。
If S5 115U, then change to OB 100	2 番目	S5 内の起動 OB 21 は自動的に OB 101 に変換されます。	S5 プログラムが S5 115U で動作していた場合は、OB 101 を OB 100 に変更する必要があります。
Note block numbers may be changed	2 番目	間接ブロック呼び出しで新ブロック番号が考慮されていません（対応するメモリまたはデータワードから取出されます）。	S5 内のロジックを変更するか、固定されたブロック呼び出しを使用します。
OB was interpreted as OB 34 from S5 115U	2 番目	使用する CPU によって、OB 34 は異なる意味を持ちます。	OB が目的のプログラムと一致しているかチェックします。
OB 23 and OB 24 have been converted to OB 122	2 番目	OB 23 と OB 24 は両方とも、S7 の OB 122 に置き換えられます。	OB 23 と 24 の内容を OB 122 に変換し、他の OB 122 を削除します。
Output not allowed (GRAPH5 block)	1 番目	GRAPH 5 ブロックは変換できません。	GRAPH 7 ブロックを挿入します。
Output not allowed (product no.)	1 番目	S5 標準ファンクションブロックは、S7 FC と置き換える必要があります。	なし。
Please check time interval settings	2 番目	S7 では時間間隔を S5 よりも正確に設定することができます。	「ハードウェアのコンフィグレーション」機能を使って時間間隔を調整します。
Please observe different STOP commands	2 番目	STP、STS、STW が区別されていません。	プログラムファイルをチェックします。
Preheader not available	1 番目	FB と FX については、ジャンプラベルの識別子が見つかりません。DB と DX については、データフォーマットが見つかりません。	別のファイル内にプリヘッダが存在するかどうかをチェックします。
RLO is set	2 番目	S5 命令 SU と RU が使用されている場合、S7 では RLO が設定されます。	必要に応じて CLEAR 命令を追加します。
S5 screen DB was not used to assign parameters to S7	1 番目	MASK が、DW0 と DW1 内にあります。	STEP 7 を使用しパラメータをプログラミング装置に割付ます
System preferences cannot be set by the S5/S7 Converter	2 番目	DB と DX は変換されますがS5 の場合とは意味が異なります。	コンフィグレーションテーブルを使用しシステム設定。

変換したプログラムの編集

7

編集の準備

変換中に作成された STL ソースファイルを編集するには、次の準備作業が必要です。

- 変換中に生成されたメッセージをプリントアウトする
- SIMATIC マネージャを使ってプロジェクト内に S7 プログラムを作成する（作成していない場合）
- メニューコマンド[**挿入 ▸ 外部ソースファイル**] を使って、STL ソースファイルプログラムをこの S7 プログラムの “Source Files” コンテナにインポートする
- 変換したファイルを開く

ファイルの編集

生成された STL ソースファイルを編集するには、次の手順を実行することをお勧めします。

- 対話式モードでプログラムを編集するが、警告に基づいて変換されなかった S5 命令とオーガニゼーションブロックについては修正または追加を行う（第1部を参照）。

7.1 アドレスの変更

入出力モジュールは通常、アドレス変更の影響を受けます。これらのモジュールのアドレスは HWConfig で確認できます。

7.1.1 アドレス指定変更のオプション

S5 の再配線

変換前に、「再配線」機能を使って、S5 の各アドレスを S7 の新しいアドレスに適合させることができます。

S7 の再配線

SIMATIC マネージャには、ブロックの再配線をソースファイルから自動的に実行する機能があります。

ブロックを再配線するには、次の手順に従います。

1. SIMATIC マネージャで、プログラム内の再配線するブロックを選択します。
2. メニューコマンド [**オプション ▸ 再配線**] を選択して、再配線に使用するテーブルを開きます。
3. テーブル内の各アドレスに古いアドレスと新しいアドレスを入力し、それから保存します。

これで、変更されたアドレスがブロックに格納されます。

S7 ソースファイルのアドレスの変更

プログラムで、S7 の入力/出力へのアクセスおよび新しいﾓｼﾞｭｰﾙアドレスへの直接 I/O アクセスを調整します。

S7 ソースファイルでは、メニューコマンド [**編集 ▸ 置換**] を選択することにより、絶対アドレスを簡単に変更できます。

注意: 古いアドレス領域と新しいアドレス領域が重複する場合、意図しない方法で変更される場合があります。

（シンボルアドレス指定で）新規 S7 ソースファイルを生成

シンボルアドレス指定を使用する場合、シンボルテーブルを使って再配線を行うこともできます。

前提条件

再配線を行う前に、コンパイル済みのプログラムと、絶対アドレスの変更に必要な全シンボルが格納されたシンボルテーブルが揃っていなければなりません。

手順

アドレスを変更するには、次の手順に従います。

1. 変更するアドレスが格納されたブロックを開きます。メニューコマンド [**オプション ▸ カスタマイズ**] を選択してダイアログボックスを表示し、[編集] タブで [シンボル表示] を選択します。
   変更したアドレスをもつ全ブロックに対して、この手順を繰り返します。
2. メニューコマンド [**ファイル ▸ ソースファイルの生成**] を選択して、ブロックからソースファイルを生成します。ソースファイルの名前を入力したら、ダイアログボックスでブロックを選択できます。

連続するブロックを作成する場合は、呼び出し階層を考慮に入れてください。一般に、呼び出されるブロックはすでに存在していなければなりません。つまり、呼び出されるブロックは、ソースファイルで呼び出し元ブロックの前に入力しなければなりません。

結果: 生成されたソースファイルには、シンボルアドレス指定による命令が格納されます。

3. ここで、シンボルテーブルで再配線を実行することができます。変更された S5 アドレスを新しい S7 アドレスで置換します。
4. ソースファイルをコンパイルすると、ブロックに新しいアドレスが保存されます。

7.2 変換できないファンクション

自動的に変換できないアドレスおよび命令は、生成された S7 プログラムにコメントとして保存されます。これらは、ユーザ自身が変換しなければなりません。

これらの命令は、2通りの方法で変換できます。

- これらの命令に対して、ユーザ自身の S7 STL 命令シーケンス（マクロ）を定義する（命令がユーザプログラムで発生する場合）。その後、変換中に使用することができる
- 生成され S7 プログラムで命令シーケンスを編集する

どちらの方法が適しているかは、ユーザプログラムでこの種の命令が発生する回数によって異なります。

変換できないアドレスおよび命令については、第3章11節および第3章12節で調べることができます。この節には、変換できないファンクションを S7 で作成する際の留意点も記載されています。

7.3 間接アドレス指定 - 変換

S5/S7 コンバータでは、DO FW および DO DW による間接アドレス指定の変換に STEP 7 命令を使用します。生成される命令シーケンスは、一般に非常に大きくなります。これは、STEP 5 ポインタを STEP 7 フォーマットに変換しなければならないためです。このため、変換時には、アキュムレータの内容とステータスワードをバッファに保存しておく必要があります。

プログラムで間接アドレス指定を頻繁に行う場合は、STEP 7 の間接アドレス指定に適合させるとよいでしょう。適切なプログラミング技術を使えば、メモリ空間は大幅に節約することができます。

次のリストでは、S5/S7 コンバータが間接アドレス指定を変換する方法をケース別に説明しています。

タイマおよびカウンタ

タイマおよびカウンタの間接アドレス指定は、テンポラリローカルデータワードによりメモリ間接アドレス指定に変換されます。

ブロック

ブロックの間接アドレス指定は、テンポラリローカルデータワードによりメモリ間接アドレス指定に変換されます。

変換中は新しいブロック番号は考慮されないので、修正する必要があります。

アドレス

アドレスの間接アドレス指定は、ビット単位およびワード単位でレジスタ間接アドレス指定に変換されます。変換には、アドレスレジスタ AR1 とテンポラリローカルデータが、ステータスワード STW、ACCU 1、ACCU 2 のバッファとして使用されます。

BR レジスタによる間接アドレス指定

命令は変換されません。間接アドレス指定は、S7 でプログラミングし直す必要があります。

間接アドレス指定のその他のタイプ

命令は S7 でプログラミングし直す必要があります。

間接アドレス指定の詳細については、第 3 章 13.4 節を参照してください。

7.4 直接メモリアクセスの使用

STEP 5 では、絶対メモリアドレスは一部のファンクションに使用されていました。この種のアクセスは STEP 7 では使用されません。

STEP 5	STEP 7
「非常に長い」データブロックにおけるデータアドレスの指定	255 を超えるデータアドレスの指定は、標準命令（L, T, ...）で可能です。
BR レジスタによる間接アドレス指定	間接アドレス指定は、レジスタ間接アドレス指定で実行できます（第3章13.4節および「*Statement List Programming Manual* **/232/**」を参照のこと）。
ブロック転送の使用	ブロック転送には、システムファンクション SFC20 BLKMOVが用意されています。コピーされるメモリ領域は、ブロックパラメータで指定します。メモリ領域が可変の場合、パラメータ “ANY Pointer” で指定できます。このパラメータには、ユーザプログラムでアクセスできます。

7.5 パラメータの割り付け

S5コマンド B<Block Parameters>

転送するブロックのタイプに応じて、ステートメント B <Formal Parameters of Type “B”> は S5 で以下のいずれかのように実行されます。

- “JU Logic Block”
- “A DB Data Block”

この場合、仮パラメータのタイプ情報がないため、自動変換はできません。パラメータタイプ “B” を使用する X 命令をプログラムで調べ、これらの命令を手動で変換します。

実パラメータ

パラメータが割り付けられたファンクションブロックでは、S5/S7 コンバータは実パラメータを変更せずにブロック呼び出しに使用します。実パラメータでアドレスを定義済みの場合、このアドレス定義をチェックし、必要に応じて変更する必要があります。

例:

- データワード番号の定義:

  これは、バイト単位のアドレス指定に変換する必要があります。

- I/O アドレスの定義:

  新しいモジュールアドレスを使用する必要があります。

- ブロックの転送:

  ブロックには新しいブロック番号を指定する必要があります。

7.6 標準ファンクション

S5標準ファンクションブロック

S5 プログラムには、標準ファンクションブロック（SFB）が含まれ、次のように示されます。

- 変換前: ダイアログボックス [S5ファイルの変換 [<Name>ST.S5D]] の [Std.] にアスタリスクが示される
- 変換後: メッセージ “Output not allowed（product no.）” が表示される

STEP 7 標準ソフトウェアには、浮動小数点演算、シグナルファンクション、内臓ファンクション、基本ロジックファンクション、数値演算ファンクション（FC61 ～ FC125）用に変換された S7 ファンクション（以前の S5 標準ファンクションブロック）が用意されています（第3章9節を参照）。

FC の組み込み

S7 ファンクションを S7 プログラムに組み込むには、次の手順に従います。

1. ファンクションを組み込みたいプロジェクトを開きます。
2. 変換された S5 ファンクション（StdLib30）を使って、SIMATIC マネージャで標準ライブラリを開きます。
3. 必要な S7 ファンクションを標準ライブラリから S7 プログラムへコピーします。

プログラムのコンパイル

8

変換と編集が終わったプログラムは、実行する前に STL コンパイラでコンパイルする必要があります。コンパイル方法は、新しく作成したテキストファイルをコンパイルする場合と同じです。

データー貫性のチェック

メニューコマンド [**ファイル▸一貫性チェック**] を選択して、ソースファイルの構文と一貫性をチェックします。このとき、ブロックは作成されません。次のような項目のチェックが行われます。

- 構文
- シンボル
- プログラム内に呼び出されたブロックがあるかどうか

チェックが完了するとコンパイラレポートが表示され、コンパイルされたファイル、コンパイルされた行数、発生したエラーと警告の数が示されます。

ソースファイルのコンパイル

メニューコマンド [**ファイル▸コンパイル**] を選択して、ソースファイルをブロックに変換します。

コンパイルが完了すると、コンパイラレポートが表示され、発生したエラーが示されます。このレポートは、ファイルの一貫性チェックの後に表示されるレポートと類似しています。ソースファイルに複数のブロックがある場合、エラーのないブロックがコンパイルされ、保存されます。

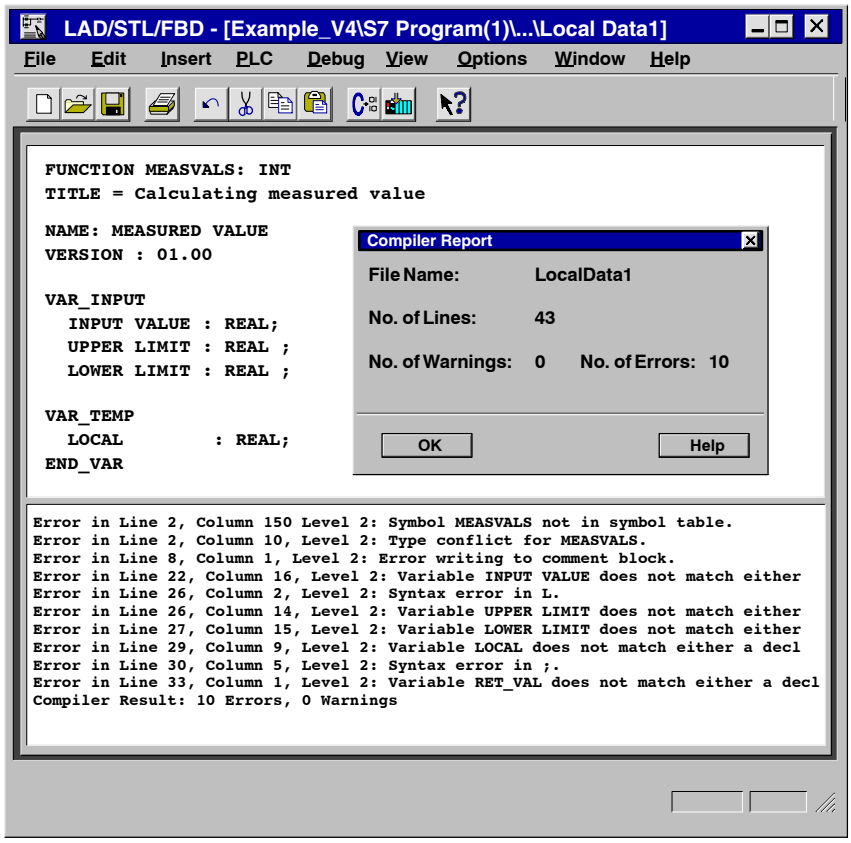

図 8-1　　ソースファイルの一貫性チェックおよびコンパイル

トラブルシューティング

一貫性チェックとコンパイルの実行後に、変換したプログラムでエラーまたは警告が発生している場合、ソースファイルのウィンドウの2番目のセクションにこのエラー/警告のリストとその原因が示されます。ここでエラーメッセージを選択すると、対応するエラーのソースファイル内での場所が表示されます。エラーメッセージとエラーの場所がわかれば、トラブルシューティングとエラーの修正を迅速に行えます。

エラー修正と変更は、上書きモードで行えます。INSERT キーを押して、挿入モードと上書きモードを切り替えます。

実例

9

この章では、4種類のオペレーションを示す実例を紹介します。これらのオペレーションは、S7 で新規のものか、S5 とは異なる方法で実行されるものです。

- アナログ値の処理
- ローカルデータ
- オーガニゼーションブロックの開始情報の評価
- ブロック転送

この例では、左右に回転するモータをデジタル I/O モジュールを使って制御します。速度は、アナログ入力モジュールで読み取り、アナログ出力モジュールで出力できます。この例で使用するデジタルモジュールとアナログモジュールでは、診断割り込みを起動できなければなりません。

コンフィグレーション

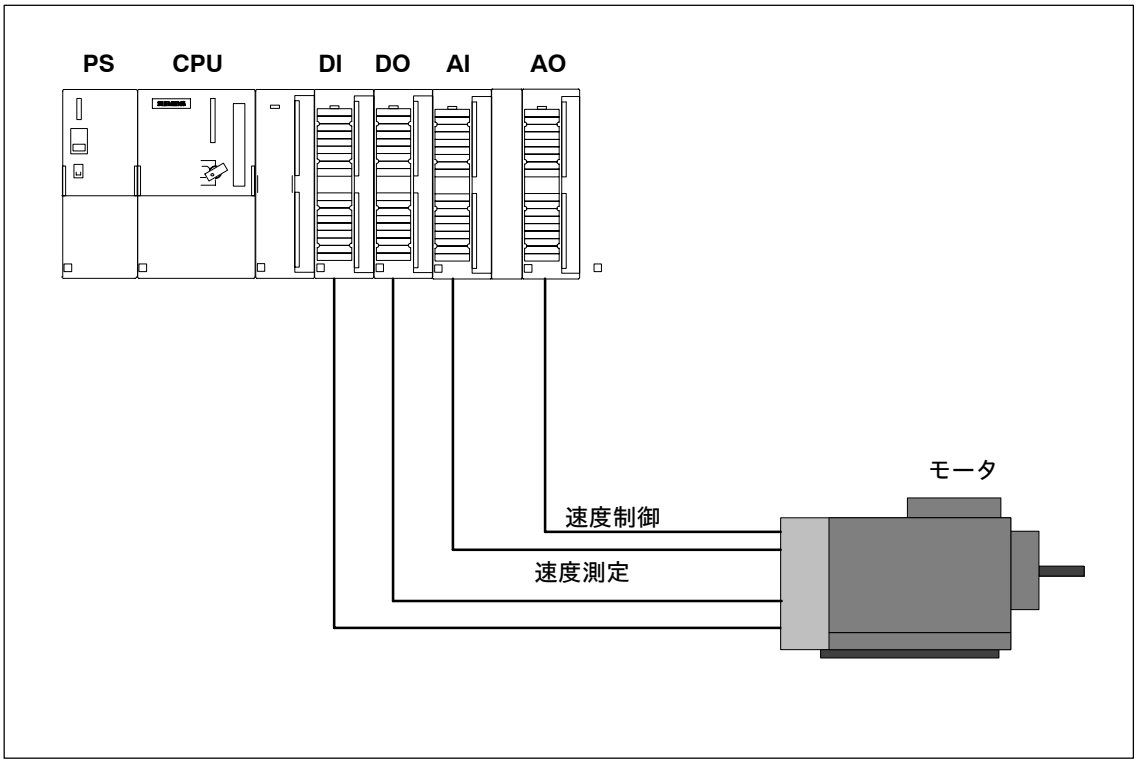

図 9-1　　実例のコンフィグレーション

9.1 アナログ値の処理

アナログ値の変換

アナログ値は、CPU によりデジタル形式で処理されます。

アナログ入力モジュールは、アナログプロセス信号をデジタルに変換します。

アナログ出力モジュールは、デジタル出力値をアナログ信号に変換します。

S5のアナログ値表現

表 9-1　　アナログ入力モジュール 6ES5 460-7LA13 の例

分解能	アナログ値															
ビット番号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ビット値	PS	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	A	E	O

アナログ出力モジュールの値は、12 ビットの符号反転で表現されます。

アナログ入力モジュールは、必要に応じて、値を符号付12ビット値または13ビット符号反転として評価できます。

“O” ビットはオーバーフローの量を示します。

“E” ビットはエラービットで、エラーが発生すると設定されます（たとえば、パラメータが割り付けられている場合、断線など）。

“A” ビットはアクティビティビットに相当します。このビットが “0” の場合、表示された値は有効です。

S7のアナログ値表現

同じ公称レンジであれば、デジタル化されたアナログ値は入力値に対しても出力値に対しても同じです。

アナログ値は、符号反転として表現されます。

表 9-2　　S7 のアナログ入力モジュールの例

分解能	アナログ値															
ビット数	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ビット値	S	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

先頭に符号（S = 符号）のついたアナログ値は常にビット15に示されます。ここでは、“0” は正数を、“1” は負数を示します。

S7 にはエラービットはありません。

エラーが発生すると、値 W#16#7FFF が出力されます。

この場合、診断機能をもつブロックは診断割り込みを起動できます。この割り込みのパラメータは HWConfig で設定します。

アナログモジュールの分解能が 15 ビット未満の場合、アナログ値はユーザデータ内で左揃えになります。使用されていない低い値は信号状態が “0” になります。

例

この例では、モータの速度は、分解能14ビットのアナログ入力モジュールで読み取ります。この測定値は双極性（たとえば +/- 10V）になります。

上限値と下限値はパラメータとして転送されます。

アナログ値に対しては、上限値と下限値のチェックが行われます。読み取られた値が許容範囲を超えている場合、バイナリリザルト(BR = 0）でエラーが報告され、値 “0” が出力されます。値が許容範囲であれば、その値が出力されます。

アナログ値は、ファンクションのリターン値 RET_VAL を介して出力されます。この RET_VAL は機能的な値です。S7 では、S5 に比べて新しい機能が追加されています。

```
FUNCTION FC1: REAL
TITLE = Analog Value Processing
NAME:         ANALOG
VERSION:      01.00
VAR_INPUT
       INPUT VALUE   : INT;        // Input value
       UPPER LIMIT   : REAL;       // Upper limit for the analog value
       LOWER LIMIT   : REAL;       // Lower limit for the analog value
END_VAR

BEGIN
NETWORK
TITLE = Checking Upper and Lower Limits
       O(;
       L      INPUT VALUE;  // Input value > Upper limit
       L      +27648;
       >I;
       );
       O(;                  // or
       L      INPUT VALUE;  // Input value < Lower limit
       L      -27648;
       <I;
       );
       NOT;
       L      0;
       JNB    END;          // If upper or lower limit exceeded, no further
                            // processing, return value = 0 and BR = "0"
                            // If no upper or lower limt exceeded => BR = "1"
NETWORK
TITLE = Converting Digital Value into Revolutions
       L      UPPER LIMIT;  // Formula for converting INPUT VALUE into
                            // revolutions:
       L      LOWER LIMIT;  // Analog value = (UPPER LIMIT - LOWER LIMIT)
                            //              * INPUT VALUE
       -R;                  //              / (55296 (number of units))
       L      INPUT VALUE;
       ITD;                 // Convert value into floating-point number
       DTR;
       *R;
       L      55296.0;
       /R;
END:   T      RET_VAL;

       BE;

END_FUNCTION
```

図 9-2　　アナログ値の処理

9.2 テンポラリローカルデータ

テンポラリローカルデータは、バッファ領域として機能し、S5 で使用されているスクラッチパッドフラグに代わるものです。テンポラリローカルデータは、すべてのロジックブロックで使用できます。これらのデータはローカルデータスタック（L スタック）に置かれ、ロジックブロックの処理が完了すると消去されます。

例 1

この最初の例では、**シンボル**によりアドレス指定されるテンポラリローカルデータをバッファとして使用します。設定済みの速度は、分解能14ビットのアナログ出力モジュールに合わせてデジタル値に変換されます。この測定値は双極性（たとえば +/-10V）です。

上限値と下限値はパラメータとして転送されます。

パラメータ値は、ファンクションのリターン値T（RET_VAL）を介して出力されます。各ファンクションには、オプションとしてリターン値を供給することができます。リターン値のデータタイプは、ファンクション記述で示されます。リターン値が供給されないと、データタイプの位置にはVOID と示されます。

```
FUNCTION FC2: INT
TITLE = Calculating Measured Value
NAME:         MEASURED VALUE
VERSION:      01.00
VAR_INPUT
       INPUT VALUE   : REAL;       // Input value (current value)
       UPPER LIMIT   : REAL;       // Upper limit
       LOWER LIMIT   : REAL;       // Lower limit
END_VAR

VAR_TEMP
       LOCAL         : REAL;       // Local data as intermediate result
END_VAR
BEGIN
NETWORK
TITLE = Calculating Measured Value
       L      INPUT VALUE;         // Formula for calculating units:
       L      55296.0;             // Measured value = INPUT VALUE
       *R;                         //             * 55296 (number of units)
                                   //             / (UPPER LIMIT - LOWER LIMIT)
       T      LOCAL;               // Intermediate result in local data
       L      UPPER LIMIT;         // Buffer
       L      LOWER LIMIT;
       -R;
       L      LOCAL;
       TAK;
       /R;
       RND;                        // Convert floating-point number into integer
       T      RET_VAL;

END_FUNCTION
```

図 9-3　　測定値の計算

例 2

2番目の例では、**絶対**アドレス指定が行われるローカルデータ（たとえばS5 スクラッチパッドフラグ）を使用し、モータの左右回転を制御する方法を示します。この例では、入力バイトと出力バイトはローカルデータ領域にコピーされます。L スタックは TLAD/STL/FBD エディタで使用されるので、ユーザはローカルスタックの一部をテンポラリローカルデータ用に確保しておかなければなりません。ローカルデータの絶対アドレスは、宣言セクション内のブロックで読むことができます。ローカルデータビット同士は、プログラムの論理演算によりリンクされています。これにより、出力信号が生成され、ブロックの最後で、ローカルデータから出力バイトへと書き込まれます。入力バイトと出力バイトのアドレスにはパラメータを割り付けることができます。

注

既存のローカルデータの前に新しい変数を挿入すると、次のローカルデータアドレスにシフトされます。

表 9-3　　入力、出力、およびローカルデータの割り付け

アドレス	ローカルデータ	名称	説明
I n.0	L 0.0	ON	ON スイッチ
I n.1	L 0.1	STOP	モータ停止
I n.2	L 0.2	EMERGENCY_STOP	緊急停止ボタン
I n.3	L 0.3	MOTOR_RIGHT	モータ: 右回転
I n.4	L 0.4	MOTOR_LEFT	モータ: 左回転
I n.5	L 0.5	LIMIT SWITCH RIGHT	リミットスイッチ, 右
I n.6	L 0.6	LIMIT SWITCH LEFT	リミットスイッチ, 左
I n.7	L 0.7	-	未使用
Q m.0	L 1.0	READY	モータが動作可能
Q m.1	L 1.1	CLOCKWISE	右回転アクティブ
Q m.2	L 1.2	COUNTER-CLOCKWISE	左回転アクティブ
Q m.3	L 1.3	POSITION REACHED	到達位置

運転

ON スイッチで電圧がかかります。モータは動作可能状態になっています。この状態は、出力 READY によって示されます。モータは、MOTOR_RIGHT ボタンで右回転し、MOTOR_LEFT ボタンで左回転します。一度の操作で左右どちらかでしか運転しません。モータは、回転方向を変更する前に STOP スイッチで休止させる必要があります。移動制限スイッチに達すると、モータは停止します。モータはEMERGENCY_STOP ボタンでも停止します。EMERGENCY_STOP ボタンで停止すると、このボタンをリセットしてからでなければモータを再始動できません。

```
FUNCTION FC3: VOID
TITLE = Motor Control
NAME:         MOTOR
VERSION:      01.00

VAR_INPUT
       INPUT BYTE    : BYTE;       // Input byte
END_VAR
VAR_IN_OUT
       OUTPUT BYTE   : BYTE;       // Output byte
END_VAR
VAR_TEMP
       IMAGE_INPUT BYTE     : BYTE;// Image of input byte
       IMAGE_OUTPUT BYTE    : BYTE;// Image of output byte
END_VAR

BEGIN
NETWORK
TITLE =  Motor Control
       L      INPUT BYTE;          // Copy input byte into local data area
       T      IMAGE_INPUT BYTE;
       L      OUTPUT BYTE;         // Copy output byte into local data area
       T      IMAGE_OUTPUT BYTE;
       ON     L0.0;                // Motor not switched on (no voltage)
       ON     L0.2;                // or EMERGENCY_STOP button pushed
       R      L1.0;                // => Motor is ready to reset
       R      L1.1;                // => Reset motor control
       R      L1.2;
       R      L1.3;                // => Reset position reached
       JC     END;                 // => No further signal evaluation
       A      L0.0;                // Motor switched on
       S      L1.0;                // => Set motor switched on
       A      L0.3;                // Operate motor clockwise
       AN     L0.4;                // Disable: no operation counter-clockwise
       AN     L1.2;                // and counter-clockwise not active
       FP     M0.0;                // Create positive edge
       S      L1.1;                // Then: switch on clockwise
       R      L1.3;                // Reset position reached
       A      L0.4;                // Operate motor counter-clockwise
       AN     L0.3;                // Disable: no operation clockwise
       AN     L1.1;                // and clockwise not active
       FP     M0.1;                // Create positive edge
       S      L1.2;                // Then: switch on counter-clockwise
       R      L1.3;                // Reset position reached
       O(;
       A      L0.5;                // Right limit switch reached and
       A      L1.1;                // clockwise active
       );
       O(;                         // or
       A      L0.6;                // Left limit switch reached and
       A      L1.2;                // counter-clockwise active
       );
       S      L1.3;                // => Position set reached
       O      L0.1;                // Stop motor switch pushed or
       O      L1.3;                // position reached
       R      L1.1;                // => Reset motor operation
       R      L1.2;
END:   L      IMAGE_OUTPUT BYTE;   // Copy local data to output byte
       T      OUTPUT BYTE;
END_FUNCTION
```

図 9-4　　モータ制御ファンクション

9.3 診断割り込み OB（OB82）による開始情報の評価

開始情報

オペレーティングシステムによってオーガニゼーションブロックが呼び出されると、ローカルデータスタックにシステム関連の開始情報が示されます。この情報は20バイト長で、OB 処理の開始後に使用できるようになります。

OB82のスタート情報

診断割り込み OB の開始情報には、4バイトの診断情報をもつ論理ベースアドレスが含まれています。この開始情報の正確な構造については、「*Reference Manual* **/235/**」で説明しています。対応する変数宣言テーブルのテンプレートは、“StdLib30” 標準ライブラリの “StdLib30” にあります。

HWConfig で前にコンフィグレーションした診断割り込みパラメータに基づき、診断モジュールは、CPU に診断割り込みを求める要求を出します。このファンクションは、着信イベントと発信イベントの両方に適用されます。この要求の発行後、オペレーティングシステムによりオーガニゼーションブロック OB82 が呼び出されます。

診断割り込み OB の呼び出しを無効にしたり、遅延させたり、再度有効にするには、システムファンクション（SFC）39 ～ 42 を使用します。詳細については、「*Reference Manual* **/235/**」を参照してください。

例

次のサンプルプログラムは、外部補助電圧を評価する方法を示しています。外部補助電圧に割り込みがかかると、DB82 “DB_DIAG” にビット NO_EXT_VOLTAGE が設定されます。さらに、モジュールアドレスとイベントの時間が保存されます。この情報は、プログラムで後で処理することができます。

Before the STL ソースファイルがコンパイルされる前に、データブロック DB82 “DB_DIAG” のシンボルをシンボルテーブルに入力する必要があります。

```
DATA_BLOCK DB_DIAG
TITLE = Diagnostic Data
NAME:         DB_DIAG
VERSION:      01.00

STRUCT
       MDL_ADDR              : INT;         // Module address
       NO_EXT_VOLTAGE        : BOOL;        // No error bit for ext. aux. voltage
       DATE_TIME     : DATE_AND_TIME;       // Date and time at which the
                                            // diagnostic interrupt was triggered
       SFC_RET_VAL           : INT;         // Return code of SFC BLKMOV
END_STRUCT;

BEGIN
END_DATA_BLOCK

ORGANIZATION_BLOCK OB82
TITLE = Diagnostic Interrupt
NAME:         Diagnostic
VERSION:      01.00
VAR_TEMP
       OB82_EV_CLASS         : BYTE;// Event class and IDs:
                                    // B#16#38: outgoing event
                                    // B#16#39: incoming event
       OB82_FLT_ID           : BYTE;// Error code (B#16#42)
       OB82_PRIORITY         : BYTE;// Priority class 26 or 28
       OB82_OB_NUMBR         : BYTE;// OB number
       OB82_RESERVED_1       : BYTE;// Reserved
       OB82_IO_FLAG          : BYTE;// Input module: B#16#54
                                    // Output module: B#16#55
       OB82_MDL_ADDR         : INT; // Logical base address of module
                                    // where the fault occurred
       OB82_MDL_DEFECT       : BOOL;// Module is defective
       OB82_INT_FAULT        : BOOL;// Internal fault
       OB82_EXT_FAULT        : BOOL;// External fault
       OB82_PNT_INFO         : BOOL;// Channel fault
       OB82_EXT_VOLTAGE      : BOOL;// External voltage failed
       OB82_FLD_CONNCTR      : BOOL;// Front panel connector not plugged
       OB82_NO_CONFIG        : BOOL;// Module is not configured
       OB82_CONFIG_ERR       : BOOL;// Incorrect parameters on module
       OB82_MDL_TYPE         : BYTE;// Bit 0 to 3: Module class
                                    // Bit 4: Channel information exists
                                    // Bit 5: User information exists
                                    // Bit 6: Diag. interrupt from substitute
                                    // Bit 7: Reserve
       OB82_SUB_MDL_ERR      : BOOL;// Submodule is missing or has an error
       OB82_COMM_FAULT       : BOOL;// Communication problem
       OB82_MDL_STOP : BOOL;        // Operating mode (0: RUN, 1: STOP)
       OB82_WTCH_DOG_FLT     : BOOL;// Watchdog timer responded
       OB82_INT_PS_FLT       : BOOL;// Internal power supply failed
       OB82_PRIM_BATT_FLT    : BOOL;// Battery dead
       OB82_BCKUP_BATT_FLT   : BOOL;// Entire backup failed
       OB82_RESERVED_2       : BOOL;// Reserved
       OB82_RACK_FLT         : BOOL;// Rack failure
       OB82_PROC_FLT         : BOOL;// Processor failure
       OB82_EPROM_FLT        : BOOL;// EPROM fault
       OB82_RAM_FLT          : BOOL;// RAM fault
```

continued

図 9-5　　診断データの評価

```
       OB82_ADC_FLT         : BOOL;       // ADC/DAC error
       OB82_FUSE_FLT        : BOOL;       // Fuse blown
       OB82_HW_INTR_FLT     : BOOL;       // Hardware interrupt lost
       OB82_RESERVED_3      : BOOL;       // Reserved
       OB82_DATE_TIME: DATE_AND_TIME;     // Date and time when OB was called
END_VAR

BEGIN
NETWORK
TITLE = Diagnostic Interrupt
       L      OB82_MDL_ADDR;              // Save module address
       T      DB_DIAG.MDL_ADDR;
       L      OB82_EV_CLASS;              // Event class = B#16#38:
       L      B#16#38;                    // Outgoing event
       ==I;
       JC     GO;
                                          // Incoming event:
       A      OB82_EXT_VOLTAGE;           // Check if no ext. auxiliary voltage
       S      DB_DIAG.NO_EXT_VOLTAGE;     // Set bit
       JU     TIME;
                                          // Outgoing event:
GO:    A      OB82_EXT_VOLTAGE;           // Ext. auxiliary voltage present
again
       R      DB_DIAG.NO_EXT_VOLTAGE;     // Reset bit

NETWORK
TITLE = Save Time
TIME:  CALL   SFC20(                      // SFC BLKMOV
       SRCBLK :=OB82_DATE_TIME,           // Save date and time at which
       RET_VAL:=DB_DIAG.SFC_RET_VAL,      // diagnostic interrupt
       DSTBLK :=DB_DIAG.DATE_TIME);       // was requested
END_ORGANIZATION_BLOCK
```

図 9-6　　診断データの評価、続き

9.4 ブロック転送

システムファンクション SFC20 “BLKMOV”（ブロック転送）により、メモリ領域の1つである「ソースフィールド」の内容を他のメモリ領域である「ターゲットフィールド」にコピーできます。

SFC20 “BLKMOV” により、すべての入力、出力、ビットメモリ、およびデータをコピーできます。

パラメータ

パラメータ	宣言	データタイプ	メモリ領域	説明
SRCBLK	INPUT	ANY	I, Q, M, D, L	コピーするメモリ領域（ソースフィールド）を示す。
RET_VAL	OUTPUT	INT	I, Q, M, D, L	ファンクションの処理中にエラーが発生すると、リターン値にエラーコードが示される。
DSTBLK	OUTPUT	ANY	I, Q, M, D, L	データのコピー先となるメモリ領域（ターゲットフィールド）を示す。

注

ソースフィールドとターゲットフィールドは重複しないようにします。指定したターゲットフィールドがソースフィールドよりも大きい場合、ソースフィールドのデータのみがターゲットフィールドにコピーされます。

指定したターゲットフィールドがソースフィールドよりも小さい場合、ターゲットフィールドに格納できる量のデータのみがコピーされます。

SFC20 “BLKMOV” のソース領域とターゲット領域のパラメータに定数ポインタではなく変数値を指定したい場合は、データタイプ ANY のテンポラリ変数を使って指定します。

データタイプのANYポインタの構造　次の表に、ANY ポインタの構造を示します。

表 9-4　ANY ポインタ

バイトn	バイト n+1	バイト n+2	バイト n+3	バイト n+4	バイト n+5	バイト n+6	バイト n+7	バイト n+8	バイト n+9
B#16#10	タイプ（表9-5を参照）	長さ		データブロックのデータブロック番号		領域ポインタ（図9-7を参照）			

表 9-5　タイプ（バイト n+1）

値:	01	02	03	04	05	06	07
タイプ:	BOOL	BYTE	CHAR	WORD	INT	DWORD	DINT
値:	08	09	0A	0B	0C	0E	13
タイプ:	REAL	DATE	TOD	TIME	S5TIME	DT	String

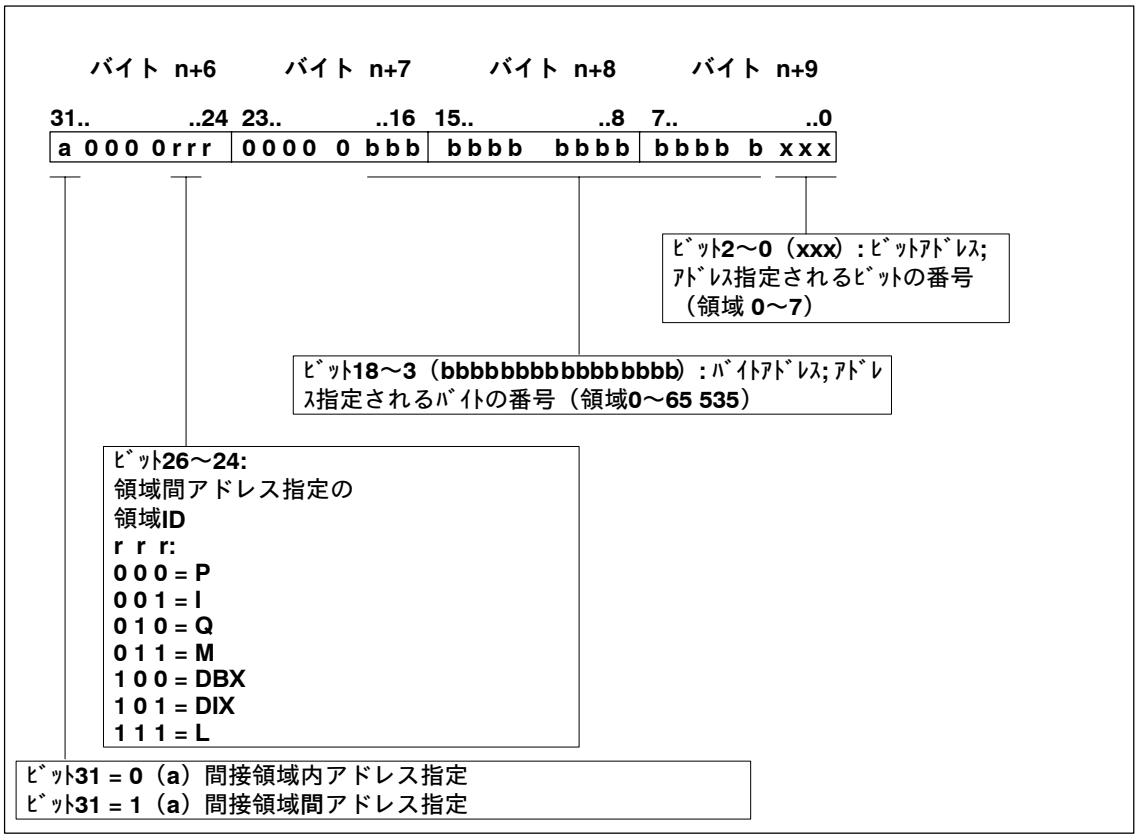

図 9-7　領域ポインタ（バイト n+6～n+9）

例

この例では、データ領域（データブロック内）のコピーにシステムファンクション SFC20 “BLKMOV” を使用するファンクションを示します。可変のソース領域とターゲット領域はパラメータとして入力できます。

原理

このファンクションでは、ローカルデータ領域に2つの ANY ポインタとターゲット領域用 ANY ポインタ1つが含まれています。一般に、ANY データタイプは、ローカルデータ領域内の変数に対して使用できます。

このファンクションでは、前述した構造で示されるように、ANY ポインタに値が割り付けられます。この値は、SFC20 “BLKMOV” が呼び出されるときにパラメータに示されます。

```
FUNCTION FC4: INT
TITLE = Copying Data Areas
NAME:         COPY
VERSION:      01.00

VAR_INPUT
       SOURCE_DBNO   : INT;        // DB no. of source area
       SOURCE_BEGIN  : INT;        // Data word no. of beginning of source area
       SOURCE_LENGTH : INT;        // Length of source area in bytes
       DEST_DBNO     : INT;        // DB no. of destination area
       DEST_BEGIN    : INT;        // Data word no. of beginning of dest. area
       DEST_LENGTH   : INT;        // Length of destination area in bytes
END_VAR

VAR_TEMP
       POINTER_SOURCE: ANY;        // ANY pointer for the source area
       POINTER_DEST  : ANY;        // ANY pointer for the destination area
END_VAR

BEGIN
NETWORK
TITLE = Preparing Source Pointer
       L      P##POINTER_SOURCE;   // Load address of pointer in source area
       LAR1;                       // into address register 1
       L      W#16#1002;           // Write area ID for data area in
       T      LW[AR1, P#0.0];      // ANY pointer for source
       L      SOURCE_DBNO;         // Write DB no. in ANY pointer for source
       T      LW[AR1, P#4.0];
       L      SOURCE_BEGIN;        // Convert beginning of data area
       SLD    3;                   // into pointer format,
       OD     DW#16#84000000;      // Link area ID
       T      LD[AR1, P#6.0];      // and write in ANY pointer for source
       L      SOURCE_LENGTH;       // Write length of data area in ANY pointer
       T      LW[AR1, P#2.0];      // for source
                                                          continued
```

図 9-8　　データ領域のコピー

```
NETWORK
TITLE = Preparing Destination Pointer
       L      P##POINTER_DEST;     // Load address of pointer to dest. area
       LAR1;                       // in address register 1
       L      W#16#1002;           // Write area ID for data area in
       T      LW[AR1, P#0.0];      // ANY pointer for destination
       L      DEST_DBNO;           // DB no. in ANY pointer for destination
       T      LW[AR1, P#4.0];
       L      DEST_BEGIN;          // Convert beginning of data area
       SLD    3;                   // into pointer format
       OD     DW#16#84000000;      // Link area ID
       T      LD[AR1, P#6.0];      // and write in ANY pointer for destination
       L      DEST_LENGTH;         // Write length of data area to ANY pointer
       T      LW[AR1, P#2.0];      // for destination
NETWORK
TITLE = Copying Data
       CALL   SFC 20(              // Copy data with SFC BLKMOV (block transfer)
       SRCBLK := POINTER_SOURCE,   // Pointer to source area
       RET_VAL:= RET_VAL,          // Return code of SFC BLKMOV
       DSTBLK := POINTER_DEST);    // Pointer to destination area
END_FUNCTION
```

図 9-9　　データ領域のコピー、続き

9.5　サンプルの呼び出し

この節では、シンボルテーブル、ブロックパラメータへの値の割り付けに必要なデータブロック、前述したファンクションの呼び出しをもつオーガニゼーションブロック OB1 を説明します。

表 9-6　　シンボルテーブル

シンボル	アドレス	データタイプ	コメント
DB_DIAG	DB82	DB82	診断データブロック
DB_MEASVALS	DB100	DB100	測定値用のデータブロック
DB_MOTOR_1	DB110	DB110	モータ1のデータブロック
ERROR	MW 100	WORD	ブロック転送用の FC4 ファンクションのリターン値

```
DATA_BLOCK DB_MEASVALS
TITLE = Measured Values
NAME:         DB_MEASVALS
VERSION:      01.00
STRUCT
       ANALOGVAL_1   : REAL;       // Analog value 1 from FC1
       ANALOGVAL_2   : REAL;       // Analog value 2 from FC2
       DIGITALVAL_2  : INT;        // Digitalized measured value from FC2
END_STRUCT;
BEGIN
END_DATA_BLOCK

DATA_BLOCK DB_MOTOR_1
TITLE = Motor Data
NAME:         DB_MOTOR_1
VERSION:      01.00
STRUCT
       CONTROL WORD  : WORD;       // Control of motor 1
       SPEED         : REAL;       // Speed of motor 1
       TEMPERATURE   : REAL;       // Temperature of motor 1
       CURRENT       : REAL;       // Current consumption of motor 1
END_STRUCT;
BEGIN
END_DATA_BLOCK

ORGANIZATION_BLOCK OB1
TITLE = Call in Cycle
NAME:         CYCLE
VERSION:      01.00
VAR_TEMP
  STARTINFO: ARRAY [1..20] of BYTE;
END_VAR
BEGIN
NETWORK
TITLE = Call of Functions
CALL   FC1(                        // Call function for
       INPUT VALUE   := IW 0,      // analog value processing
       UPPER LIMIT   := +10.0,     // Measured range: +/-10V
       LOWER LIMIT   := -10.0,
       RET_VAL       := DB_MEASVALS.ANALOGVAL_1);
                                   // RET_VAL = Analog value
                                   // Call function for calculating
CALL   FC2(                        // digitalized measured value
       INPUT VALUE   := DB_MEASVALS.ANALOGVAL_2,//
       UPPER LIMIT   := +10.0,     // Measured range: +/-10V
       LOWER LIMIT   := -10.0,
       RET_VAL       := DB_MEASVALS.DIGITALVAL_2);
                                   // RET_VAL = digitalized meas. value
CALL   FC3(                        // Call function for motor control
       INPUT BYTE    := IB 4,
       OUTPUT BYTE   := QB 8);
CALL   FC4(                        // Call function for block transfer
       SOURCE_DBNO   := 100,       // Source: DB100
       SOURCE_BEGIN  := 0,         // From data byte DBB 0
       SOURCE_LENGTH := 8,         // Length: 4 Byte
       DEST_DBNO     := 110,       // Destination: DB110
       DEST_BEGIN    := 2,         // From data byte DBB 6
       DEST_LENGTH   := 8,         // Length: 4 bytes
       RET_VAL       := ERROR);    // RET_VAL = Error code for SFC20 BLKMOV
END_ORGANIZATION_BLOCK
```

図 9-10　　OB1

Teil 1: Planung des Umstiegs 付録

アドレスリストおよび命令リスト

A

A.1 アドレス

変換可能なアドレス　以下のアドレスは変換されます。

表 A-1　変換可能なアドレス

S5 STL (ドイツ)	S5 STL (国際)	S7 STL (ドイツ)	S7 STL (国際)
”A”	”Q”	”A”	”Q”
”AB”	”QB”	”AB”	”QB”
”AD”	”QD”	”AD”	”QD”
”AW”	”QW”	”AW”	”QW”
”BF”	”BN”	””	””
”D”	”D”	”DBX”	”DBX”
”DW”	”DW”	”DBW”	”DBW”
”DD”	”DD”	”DBD”	”DBD”
”DR”	”DR”	”DBB”	”DBB”
”DL”	”DL”	”DBB”	”DBB”
”E”	”I”	”E”	”I”
”EB	”IB”	”EB”	”IB”
”ED”	”ID”	”ED”	”ID”
”EW”	”IW”	”EW”	”IW”
”M”	”F”	”M”	”M”
”MB”	”FY”	”MB”	”MB”
”MD”	”FD”	”MD”	”MD”
”MW”	”FW”	”MW”	”MW”
”PW”	”PW”	”PEW/PAW”	”PIW/PQW”
”PY”	”PY”	”PEB/PAB”	”PIB/PQB”
”QB”	”OY”	”PEB/PAB”	”PIB/PQB”
”QW”	”OW”	”PEW/PAW”	”PIW/PQW”
”S”	”S”	”M”	”M”
”SD”	”SD”	”MD”	”MD”
”SW”	”SW”	”MW”	”MW”

表 A-1　　変換可能なアドレス、続き

S5 STL (ドイツ)	S5 STL (国際)	S7 STL (ドイツ)	S7 STL (国際)
”SY”	”SY”	”MB”	”MB”
”T”	”T”	”T”	”T”
”Z”	”C”	”Z”	”C”
”= <Formal parameter>”	”= <Formal parameter>”	”# <Formal parameter>”	”# <Formal parameter>”

変換できないアドレス

表A-2 に**変換できない**アドレスを示します。これらのアドレスをもつ命令は、S7 プログラムではコメントとして表示されるだけなので、編集が必要です。

表 A-2　　変換できないアドレス

S5 STL (ドイツ)	S5 STL (国際)
”A1”	”A1”
”A2”	”A2”
”BA”	”RI”
”BB”	”RJ”
”BR”	”BR”
”BS”	”RS”
”BT”	”RT”
”CB”	”CY”
”CD”	”CD”
”CW”	”CW”
”GB”	”GY”
”GD”	”GD”
”GW”	”GW”
”SA”	”SA”

A.2 命令

アドレスをもたない命令の変換

表A-3 は、S7 STL への自動変換が可能な、STL で表記されたすべての S5 命令（アドレスなし）を示したものです。

表 A-3 命令の変換（アドレスなし）

S5 STL（ドイツ）	S5 STL（国際）	S7 STL（ドイツ）	S7 STL（国際）
”AF”	”RA”	”CALL SFC42”	”CALL SFC42”
”AS”	”IA”	”CALL SFC41”	”CALL SFC41”
”BEA”	”BEU”	”BEA”	”BEU”
”BEB”	”BEC”	”BEB”	”BEC”
”+D”	”+D”	”+D”	”+D”
”-D”	”-D”	”-D”	”-D”
”!=D”	”!=D”	”==D”	”==D”
”><D”	”><D”	”<>D”	”<>D”
”>D”	”>D”	”>D”	”>D”
”>=D”	”>=D”	”>=D”	”>=D”
”<D”	”<D”	”<D”	”<D”
”<=D”	”<=D”	”<=D”	”<=D”
”DED”	”DED”	”BTD”	”BTD”
”DEF”	”DEF”	”BTI”	”BTI”
”DUD”	”DUD”	”DTB”	”DTB”
”DUF”	”DUF”	”ITB”	”ITB”
”ENT”	”ENT”	”ENT”	”ENT”
”+F”	”+F”	”+I”	”+I”
”-F”	”-F”	”-I”	”-I”
”:F”	”:F”	”/I”	”/I”
”xF”	”xF”	”*I”	”*I”
”!=F”	”!=F”	”==I”	”==I”
”><F”	”><F”	”<>I”	”<>I”
”>F”	”>F”	”>I”	”>I”
”>=F”	”>=F”	”>=I”	”>=I”
”<F”	”<F”	”<I”	”<I”
”<=F”	”<=F”	”<=I”	”<=I”
”FDG”	”FDG”	”DTR”	”DTR”
”+G”	”+G”	”+R”	”+R”
”-G”	”-G”	”-R”	”-R”
”:G”	”:G”	”/R”	”/R”
”xG”	”xG”	”*R”	”*R”

表 A-3　　命令の変換（アドレスなし）、続き

S5 STL (ドイツ)	S5 STL (国際)	S7 STL (ドイツ)	S7 STL (国際)
”!=G”	”!=G”	”==R”	”==R”
”><G”	”><G”	”<>R”	”<>R”
”>G”	”>G”	”>R”	”>R”
”>=G”	”>=G”	”>=R”	”>=R”
”<G”	”<G”	”<R”	”<R”
”<=G”	”<=G”	”<=R”	”<=R”
”GFD”	”GFD”	”RND”	”RND”
”KEW”	”CFW”	”INVI”	”INVI”
”KZD”	”CSD”	”NEGD”	”NEGD”
”KZW”	”CSW”	”NEGI”	”NEGI”
”O”	”O”	”O”	”O”
”O(”	”O(”	”O(”	”O(”
”OW”	”OW”	”OW”	”OW”
”STP”	”STP”	”CALL SFC 46”	”CALL SFC 46”
”STS”	”STS”	”CALL SFC 46”	”CALL SFC 46”
”STW”	”STW”	”CALL SFC 46”	”CALL SFC 46”
”TAK”	”TAK”	”TAK”	”TAK”
”U(”	”A(”	”U(”	”A(”
”UW”	”AW”	”UW”	”AW”
”XOW”	”XOW”	”XOW”	”XOW”
”)”	”)”	”)”	”)”
”***”	”***”	”NETWORK”	”NETWORK”

アドレス付き命令の変換

表A-4 は、S7 STL への自動変換が可能な、STL で表記されたすべての S5 命令（アドレス付き）を示したものです。

表 A-4　　命令の変換（アドレス付き）

S5 STL (ドイツ)	S5 STL (国際)	S7 STL (ドイツ)	S7 STL (国際)
”A”	”C”	”AUF”	”OPN”
”ADD BF” ”ADD DH” ”ADD KF”	”ADD BF” ”ADD DH” ”ADD KF”	”+” ”+” ”+”	”+” ”+” ”+”
”AX”	”CX”	”AUF”	”OPN”
”B”	”DO”	”Instruction sequence for indirect addressing”	”Instruction sequence for indirect addressing”
”BA”	”BA”	””	””

表 A-4　　命令の変換（アドレス付き）、続き

S5 STL (ドイツ)	S5 STL (国際)	S7 STL (ドイツ)	S7 STL (国際)
”BAB”	”DOC”	”SPB”	”JC”
”D”	”D”	”DEC”	”DEC”
”E”	”G”	”CALL SFC22”	”CALL SFC22”
”EX”	”GX”	”CALL SFC22”	”CALL SFC22”
”FR”	”FR”	”FR”	”FR”
”I”	”I”	”INC”	”INC”
”L”	”L”	”L”	”L”
”LC”	”LD”	”LC”	”LC”
”NOP”	”NOP”	”NOP”	”NOP”
”O”	”O”	”O”	”O”
”ON”	”ON”	”ON”	”ON”
”P”	”TB”	”SET; U”	”SET; A”
”PN”	”TBN”	”SET; UN”	”SET; AN”
”R”	”R”	”R”	”R”
”RB”	”RB”	”R”	”R”
”RD”	”RD”	”R”	”R”
”RLD”	”RLD”	”RLD”	”RLD”
”RLW”	”RLW”	”RLW”	”RLW”
”RRD”	”RRD”	”RRD”	”RRD”
”RRW”	”RRW”	”RRW”	”RRW”
”RU”	”RU”	”SET; R”	”SET; R”
”S”	”S”	”S”	”S”
”SA”	”SF”	”SA”	”SF”
”SAR”	”SFD”	”SA” Timer ”ZR” Zähler	”SF” Timer ”CD” Counter
”SE”	”SD”	”SE”	”SD”
”SI”	”SP”	”SI”	”SP”
”SLD”	”SLD”	”SLD”	”SLD”
”SLW”	”SLW”	”SLW”	”SLW”
”SPA”	”JU”	”SPA”	”JU”
”SPB”	”JC	”SPB”	”JC”
”SPM”	”JM”	”SPM”	”JM”
”SPN”	”JN”	”SPN”	”JCN”
”SPO”	”JO”	”SPO”	”JO”
”SPP”	”JP”	”SPP”	”JP”
”SPR”	”JUR”	”SPA”	”JU”

表 A-4　　命令の変換（アドレス付き）、続き

S5 STL（ドイツ）	S5 STL（国際）	S7 STL（ドイツ）	S7 STL（国際）
”SPS”	”JOS”	”SPS”	”JOS”
”SPZ”	”JZ”	”SPZ”	”JZ”
”SRD”	”SRD”	”SRD”	”SRD”
”SRW”	”SRW”	”SRW”	”SRW”
”SS”	”SS”	”SS”	”SS”
”SSV”	”SSU”	”SS” Timer ”ZV” Zähler	”SS” Timer ”CU” Counter
”SU”	”SU”	”SET; S”	”SET; S”
”SV”	”SE”	”SV”	”SE”
”SVD”	”SSD”	”SSD”	”SSD”
”SVW”	”SSW”	”SSI”	”SSI”
”SVZ”	”SEC”	”SV” Timer ”S” Zähler	”SE” Timer ”S” Counter
”T”	”T”	”T”	”T”
”TNB”	”TNB”	”CALL SFC20”	”CALL SFC20”
”TNW”	”TNW”	”CALL SFC20”	”CALL SFC20”
”U”	”A”	”U”	”A”
”UN”	”AN”	”UN”	”AN”
”ZR”	”CD”	”ZR”	”CD”
”ZV”	”CU”	”ZV”	”CU”
”=”	”=”	”=”	”=”

変換できない命令　　次の表に示す S5 STL 命令は自動的に変換することができません。

表 A-5　　変換できない命令

S5 STL（ドイツ）	S5 STL（国際）
”AAS”	”IAI”
”AAF”	”RAI”
”ABR”	”ABR”
”ACR”	”ACR”
”AFF”	”RAE”
”AFS”	”IAE”
”ASM”	”ASM”
”BAF”	”BAF”
”BAS”	”BAS”

表 A-5　　変換できない命令、続き

S5 STL（ドイツ）	S5 STL（国際）
”BI”（パラメータタイプ D/定数の場合にのみ変換可能）	”DI”（パラメータタイプ D/定数の場合にのみ変換可能）
”BLD”	”BLD”
”LB”	”LB”
”LD”	”LD”
”LD=<仮パラメータ>”（パラメータタイプ D/定数の場合にのみ変換可能）	”LD=<仮パラメータ>”（パラメータタイプ D/定数の場合にのみ変換可能）
”LDI”	”LDI”
”LIM”	”LIM”
”LIR”	”LIR”
”LRB”	”LRB”
”LRD”	”LRD”
”LRW”	”LRW”
”LW”	”LW”
”LW=<仮パラメータ>”（パラメータタイプ D/定数の場合にのみ変換可能）	”LW=<仮パラメータ>”（パラメータタイプ D/定数の場合にのみ変換可能）
”MA1”	”MA1”
”MAB”	”MAB”
”MAS”	”MAS”
”MBA”	”MBA”
”MBR”	”MBR”
”MBS”	”MBS”
”MSA”	”MSA”
”MSB”	”MSB”
”SEF”	”SEE”
”SES”	”SED”
”SIM”	”SIM”
”TB”	”TB”
”TDI”	”TDI”
”TIR”	”TIR”
”TSC”	”TSC”
”TSG”	”TSG”
”TRB”	”TRB”
”TRD”	”TRD”
”TRW”	”TRW”
”TW”	”TW”
”TXB”	”TXB”
”TXW”	”TXW”
”UBE”	”UBE”

関連資料

B

/21/ Technical Overview: *S7/M7 Programmable Controllers*, Distributed I/O with PROFIBUS-DP and AS-i

/30/ 入門書：*S7-300 Programmable Controller,* Quick Startt

/70/ マニュアル：*S7-300 Programmable Controller,* Hardware and Installation

/71/ リファレンスマニュアル：*S7-300, M7-300 Programmable Controllers* Module Specifications

/72/ 命令リスト：*S7-300 Programmable Controller* CPU 312 IFM, 314 IFM, 313, 314, 315-2DP

/100/ マニュアル：*S7-400/M7-400 Programmable Controllers,* Hardware and Installation

/101/ リファレンスマニュアル：*S7-400/M7-400 Programmable Controllers* Module Specifications

/102/ リファレンスガイド：*S7-400 Instruction* List, CPU 412, 413, 414, 416

/231/ ユーザマニュアル: *Standard Software for S7 and M7,* STEP 7

/232/ マニュアル: *Statement List (STL) for S7-300 and S7-400,* Programming

/233/ マニュアル: *Ladder Logic (LAD) for S7-300 and S7-400,* Programming

/234/ プログラミングマニュアル: *System Software for S7-300 and S7-400,* Program Design

/235/ リファレンスマニュアル: *System Software for S7-300 and S7-400,* System and Standard Functions

/236/ マニュアル: *Function Block Diagram (FBD) for S7-300 and S7-400,* Programming

/249/ マニュアル: *Continuous Function Chart (CFC),* Volume 2: *S7/M7*

/250/ マニュアル: *Structured Control Language (SCL) for S7-300 and S7-400,* Programming

/251/ マニュアル: *GRAPH for S7-300 and S7-400,* Programming Sequential Control Systems

/252/ マニュアル: *HiGraph for S7-300 and S7-400,*
Programming State Graphs

/253/ マニュアル: *C Programming for S7-300 and S7-400,*
Writing C Programs

/254/ マニュアル: *Continuous Function Chart* (CFC),
Volume1

/270/ マニュアル: *S7-PDIAG for S7-300 and S7-400,*
Configuring Process Diagnostics for LAD, STL, and FBD

/271/ マニュアル: *NETPRO,*
Configuring Networks

/280/ プログラミングマニュアル: *System Software for M7-300 and M7-400,* Program Design

/281/ リファレンスマニュアル: *System Software for M7-300 and M7-400,*
System and Standard Functions

/282/ ユーザマニュアル: *System Software for M7-300 and M7-400,*
Installation and Operation

/290/ ユーザマニュアル: *ProC/C++ for M7-300 and M7-400,*
Writing C Programs

/291/ ユーザマニュアル: *ProC/C++ for M7-300 and M7-400,*
Debugging C Programs

/500/ マニュアル: *SIMATIC NET,*
NCM S7 for Industrial Ethernet

/501/ マニュアル: *SIMATIC NET,*
NCM S7 for PROFIBUS

/800/ *DOCPRO*
Creating Documentation (CD only)

/801/ *TeleService for S7, C7, and M7*
Remote Maintenance for Automation Systems (CD only)

/802/ *PLC Simulation for S7-300 and S7-400* (CD only)

/803/ リファレンスマニュアル: *Standard Software for S7-300 and S7-400,*
STEP 7 Standard Functions, Part 2 (CD only)

用語解説

A

Actual Parameter（実パラメータ）

ファンクションブロック（FB）またはファンクション（FC）の呼び出し時には、仮パラメータが実パラメータに置き換えられます。たとえば、仮パラメータ “START” は、実パラメータ “I3.6” に置き換えられます。

Address（アドレス）

アドレスは、アドレス識別子と、アドレスが格納されている物理的な記憶場所から構成されます。例：入力 I12.1、メモリワード MW25、データブロック DB3。

アドレスは STEP 7 ステートメントの一部となっており、プロセッサが命令をどこで実行するかを指定します。アドレスは絶対アドレスまたはシンボルによって表すことができます。

Assigning Parameters (パラメータの割り付け)

パラメータの割り付けとは、モジュールの動作方法を設定することです。

B

Block（ブロック）

ブロックはユーザプログラムの一部となっており、機能、構造、または目的によって分類されます。STEP 7 では、次のブロックタイプを使用できます。

- ロジックブロック（FB、FC、OB、SFB、SFC）
- データブロック（DB、SDB）
- ユーザ定義データタイプ（UDT）

Block Call（ブロック呼び出し）

ブロック呼び出しは、プログラム処理中の呼び出し対象ブロックへのブランチを意味します。

Block Parameter（ブロックパラメータ）

ブロックパラメータは多目的ブロック内のトークン値で、対応するブロックが呼び出されたときに現在値が与えられます。

C

Compiler（コンパイラ）

高水準のプログラム言語で記述されたプログラムを CPU が理解できるマシンコードにコンパイルするためのコンパイラプログラムをコンパイラと呼びます。

Configuring (コンフィグレーションする)

コンフィグレーションは、プログラマブルロジックコントローラ (PLC) の各構成部品を選択・統合し、必要なソフトウェアをインストールし、そのソフトウェアを特定のタスクに適合させることです（たとえば、モジュールにパラメータを割り付ける）。

D

Data Block (DB)（データブロック）

ユーザデータを格納しておくユーザプログラム内の領域。データブロックには、すべてのロジックブロックからアクセスできる共有データブロックと、特定のファンクションブロック（FB）呼び出しに関連付けられているインスタンスデータブロックがあります。その他すべてのブロックタイプと異なり、データブロックには論理命令が格納されません。

Data, Static（スタティックデータ）

スタティックデータは、ファンクションブロック内でのみ使用できるデータで、インスタンスデータブロックに保存されます。保存されたデータは、次にファンクションが呼び出されるまで、そのまま保存されます。

Data, Temporary（テンポラリデータ）

テンポラリデータはブロックのローカルデータで、ブロックの処理中はLスタックに保存され、処理後は削除されます。

Data Type（データタイプ）

データタイプにより、変数または定数の値をユーザプログラム内でどのように使用するかを指定することができます。SIMATIC S7 のユーザは IEC 1131-3 に準拠した 2 つのデータタイプ（基本データタイプと複合データタイプ）を使用することができます。

Data Type, Complex（複合データタイプ）

複合データタイプは、ユーザがデータタイプを宣言することによって定義されます。複合データタイプは固有の名前をもたず、一度しか使用することができません。アレイとストラクチャに区別されます。*String*、*Data*、*Time* のデータタイプも複合データタイプに属しています。

DataType, Elementary（基本データタイプ）

基本データタイプは、IEC 1131-3 に従ってあらかじめ定義されているデータタイプです。たとえば、データタイプ BOOL はバイナリ変数を定義します（「ビット」）。データタイプ INT は 16 ビットの固定小数点変数を定義します。

Declaration Section（宣言セクション）

テキストエディタを使用してプログラムを生成する場合、ロジックブロックのローカルデータは宣言セクションで宣言されます。

F

Formal Parameter（仮パラメータ）

仮パラメータは、パラメータを割り付け可能なロジックブロックの「実パラメータ」に対するトークン値です。ファンクションブロックとファンクションの場合、仮パラメータはユーザによって宣言されますが、システムファンクションブロックとシステムファンクションの場合、仮パラメータはあらかじめ設定されています。

ブロックの呼び出し時に仮パラメータには実パラメータが割り付けられ、これによって呼び出されるブロックは現在値で動作します。仮パラメータはブロックのローカルデータに属し、入力パラメータ、出力パラメータ、I/O パラメータに分類されます。

Function (FC) ファンクション)

ファンクションは、国際電気標準会議の IEC 1131-3 規格に準拠し、メモリをもたないロジックブロックです。ファンクションにより、ユーザプログラムでのパラメータの受け渡しが可能になるため、計算など、頻繁に使用される複雑なファンクションのプログラミングに適しています。FC はメモリをもたないので、FC 呼び出し後すぐに計算値を処理する必要があります。

Function Block (FB)（ファンクションブロック）

ファンクションブロックは、国際電気標準会議の IEC 1131-3 規格に準拠し、スタティックデータをもつロジックブロックです。ファンクションブロックにより、ユーザプログラムでのパラメータの受け渡しが可能になるため、制御システムやオペレーティングモード選択など、頻繁に使用される複雑なファンクションのプログラミングに適しています。ファンクションブロックにはメモリとしてインスタンスデータブロックが関連付けられるので、ユーザプログラムではいつでもパラメータ（出力パラメータなど）にアクセスすることができます。

I

I/O, Distributed (DP) (リモートI/O)

リモート I/O は、基本ラックから離れた位置にあるアナログモジュールとデジタルモジュールで構成されています。リモート I/O の特徴は、省配線を目的としたモジュール型ラックシステムにあり、このおかげで、プロセスの近くに I/O モジュールを配置してコストを削減することができます。

Instance (インスタンス)

「インスタンス」はファンクションブロックの呼び出しです。インスタンスデータブロックは、この呼び出しに関連付けられています。

Instance Data Block (インスタンスデータブロック)

インスタンスデータブロックには、ファンクションブロックの仮パラメータとスタティックデータが格納されます。このブロックは、ファンクションブロック呼び出しまたはファンクションブロックの呼び出し階層に関連付けることができます。

Instruction（命令）

命令は STEP 7 ステートメントの一部であり、プロセッサの処理動作を指定します。

L

Local Data（ローカルデータ）

ローカルデータはロジックブロックに割り付けられるデータで、ロジックブロックの宣言セクションまたは変数宣言において宣言されます。ローカルデータは、仮パラメータ、スタティックデータ、またはテンポラリデータとして使用されます（ブロックによって異なる）。

Logic Block（ロジックブロック）

SIMATIC S7 では、ロジックブロックには、STEP 7 ユーザプログラムの一部が格納されています。ブロックにはこの他に、データのみが格納されるデータブロックがあります。ロジックブロックのタイプを次に示します。

- オーガニゼーションブロック（OB）
- ファンクションブロック（FB）
- ファンクション（FC）
- システムファンクションブロック（SFB）
- システムファンクション（SFC）

M

Macro（マクロ）

マクロは、実行に適した 1 つのニーモニック呼び出しにまとめられた一連の命令です。

O

Online Help（オンラインヘルプ）

STEP 7 では操作中に状況に応じたヘルプを画面に表示することができます。

Organization Block (OB)（オーガニゼーションブロック）

オーガニゼーションブロックは、CPU オペレーティングシステムとユーザプログラムとの間のインターフェースを形成しています。ユーザプログラムが処理される順序は、オーガニゼーションブロックの中で指定します。

P

Pointer（ポインタ）

ポインタは、特定の値をもつのではなく、別の変数のアドレスをもつ変数です。ポインタ命令を使用する場合は、演算子の右側のタイプと、左側のタイプが一致しなければなりません。

Programming Language（プログラム言語）

プログラム言語はユーザプログラムを作成するために使用されます。プログラムを作成するのに必要な特定のボキャブラリを、テキスト命令またはグラフィック要素の形式で提供しています。これらの命令はユーザがエディタを使用して入力します。入力された命令は、実行可能なユーザプログラムにコンパイルされます。

Project（プロジェクト）

プロジェクトは、オートメーションタスク内のすべてのオブジェクトが収められているコンテナであり、ステーション数、モジュール数、ネットワーク内でのステーションおよびモジュールの接続方法に左右されることはありません。

R

Retentive（保持）

電源異常後もデータの値が電源異常の前と同じである場合、そのデータは保持されていると言います。データは、次の２つの方法でバックアップされます。

- 電圧バックアップ
- バックアップメモリ

S

S7 Program（S7 プログラム）

S7 プログラムは、ブロック、ソースファイル、S7 プログラマブルモジュールのチャートが収められているコンテナで、シンボルテーブルも備えています。

Shared Data（共有データ）

共有データは、どのロジックブロックからもアクセスできるデータです（ファンクション（FC）、ファンクションブロック（FB）、オーガニゼーションブロック（OB））。共有データとしては、ビットメモリ（M）、入力（I）、出力（Q）、タイマ（T）、カウンタ（C）、データブロック（DB）の要素があります。共有データには絶対アドレスまたはシンボルによってアクセスすることができます。

Statement（ステートメント）

ステートメントは、テキスト言語で作成されるユーザプログラムの最小単位です。これは、プロセッサに対するコマンドです。

Statement List (STL)（ステートメントリスト）

ステートメントリストは、マシンコードとよく似たテキストプログラム言語です。

Symbol（シンボル）

シンボルは、構文規則を考慮してユーザが定義する名前です。一度定義した名前は、プログラミング、オペレーティング、およびモニタリングにおいて使用することができます（たとえば、変数、データタイプ、ジャンプラベル、ブロックとして）。

例：
アドレス：I5.0、データタイプ：BOOL、シンボル：mer_Off_Switch

共有シンボルとブロック固有のシンボルは区別されます。共有シンボルはプログラムのすべての部分で使用できます。したがって、割り付けるシンボルは、ユーザプログラム全体を通して一意的でなければなりません。ブロック固有のシンボルは、それが割り付けられているブロック内でのみ認識されます。

Symbol Table（シンボルテーブル）

共有データのアドレスとブロックに、シンボル（つまりシンボル名）を割り付ける場合に使用するテーブルです。
例：　Emer_Off（シンボル）、I1.7（アドレス）
　　Controller（シンボル）、SFB24（ブロック）

V

Variable（変数）

変数は、STEP 7 ユーザプログラムで使用できる可変内容のデータ項目を定義します。変数はアドレスとデータタイプから構成されており、シンボル名によって識別されます。

索引

て

と

に

ぬ

ね

は

ひ

ふ

へ

ほ

ま

Siemens AG
AUT E 146

Östliche Rheinbrückenstr. 50
D-76181 Karlsruhe
Federal Republic of Germany

From:
Your Name: _
Your Title: _
Company Name: _
Street: _
City, Zip Code _
Country: _
Phone: _

Please check any industry that applies to you:

❐ Automotive
❐ Chemical
❐ Electrical Machinery
❐ Food
❐ Instrument and Control
❐ Nonelectrical Machinery
❐ Petrochemical
❐ Pharmaceutical
❐ Plastic
❐ Pulp and Paper
❐ Textiles
❐ Transportation
❐ Other _ _ _ _ _ _ _ _ _ _ _

Remarks Form

Your comments and recommendations will help us to improve the quality and usefulness of our publications. Please take the first available opportunity to fill out this questionnaire and return it to Siemens.

Please give each of the following questions your own personal mark within the range from 1 (very good) to 5 (poor).

1. Do the contents meet your requirements? ☐
2. Is the information you need easy to find? ☐
3. Is the text easy to understand? ☐
4. Does the level of technical detail meet your requirements? ☐
5. Please rate the quality of the graphics/tables: ☐

Additional comments:

Seitenübersicht für

xxx/ Umsteigerhandbuch: *Von SIMATIC S5 nach SIMATIC S7*