PLCアプリケーションの開発効率化指針

2009年（平成 21年）12 月 7 日発行

社団法人日本電機工業会

PLC技術専門委員会
プログラミングツール分科会

まえがき

この資料は，PLC技術専門委員会傘下のプログラミングツール分科会の審議を経て作成した委員会資料である。この資料は，著作権法で保護対象となっている著作物である。

この資料の一部が，技術的性質をもつ特許権，出願公開後の特許出願，実用新案権，又は出願公開後の実用新案登録出願に抵触する可能性があることに注意を喚起する。社団法人日本電機工業会は，このような技術的性質をもつ特許権，出願公開後の特許出願，実用新案権，又は出願公開後の実用新案登録出願にかかわる確認について，責任をもたない。

来　歴

Ver.	日付	改訂箇所	改定内容
1.0	2009/12/7	全頁	新規作成
1.01	2010/02/12	15頁	一部の図を修正

目　次

ページ

図表番号

PLCアプリケーションの開発効率化指針

1　PLCアプリケーション開発の課題及び対策

PLCは，産業用設備及び機械への主要な適用に加え，生活関連機器，環境関連機器などに至る用途の拡大を見せており，その市場規模は，**図1**（市場規模データの1998～2008を利用）のとおり，2008年度出荷ベースで金額は約1500億円，台数は約180万台強相当である。さらに，ここ10年間ではメモリ容量も増大している。また，機械1台当たりのPLC平均装着率も増加（**表1**）しており，今後も増加する傾向である。設備の自動化計画の統計では，実績及び見込み共に規模の拡大傾向を示しており，PLCアプリケーション・ソフトウェアを開発する部門が扱うソフトウェアの物量が急速に増加している。

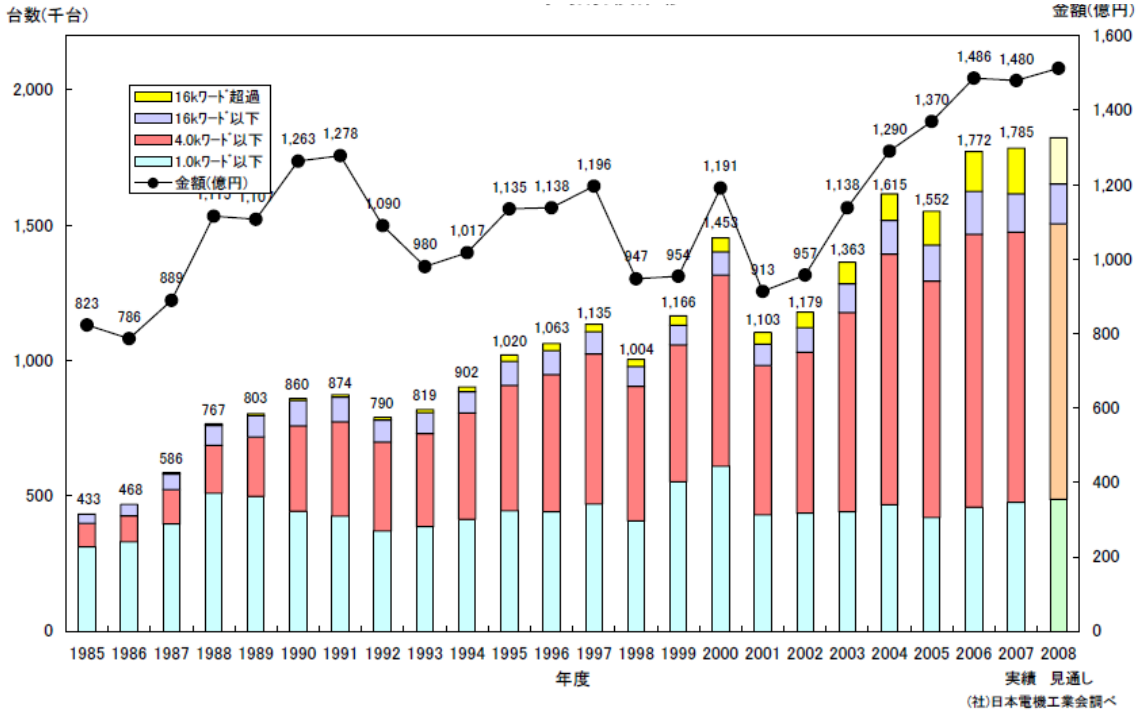


図1—PLC市場規模推移

表1—機械1台当たりのPLC平均装着率の推移

	回答事業所数	平成16年度	平成17年度	平成18年度（見込み）	平成19年度（見通し）	傾向
全体	91	0.138	0.156	0.180	0.779	↗
搬送機	20	0.456	0.473	0.480	0.516	↗
組立加工機械	11	1.147	1.006	0.927	1.033	↘↗
金属加工機械	11	1.015	1.012	1.016	1.011	→
工作機械	9	0.821	0.805	0.811	0.849	→↗
産業用ロボット	6	0.565	0.607	0.468	0.322	↘
半導体・液晶製造装置	12	1.909	1.761	1.532	1.395	→ ↘
電子部品関連機械	4	2.524	2.893	2.493	2.482	↗→
食品加工機械	6	0.524	0.526	0.536	0.527	→↗
包装機械	4	0.927	0.931	0.995	0.960	→↗
樹脂加工機械	6	1.029	1.030	1.029	1.026	→
繊維機械	4	0.052	0.054	0.070	0.063	↗→
印刷機械	4	1.216	1.306	1.267	0.867	→ ↘
木材加工機械	2	0.600	0.800	0.800	0.880	→↗
ゴム加工機械	0	－	－	－	－	
試験装置	5	0.929	0.986	0.914	0.985	↗
放送・舞台装置	0	－	－	－	－	
娯楽機械	0	－	－	－	－	
プラント制御装置	14	1.000	1.233	1.224	1.281	↘↗
受変電・空調設備	5	0.933	1.283	1.189	1.348	↗
その他	19	0.037	0.041	0.051	0.948	↗

注）平成17年度の数値は、表1と同じ（再掲）
＊対象は各機械毎に製品生産台数とＰＬＣ使用台数両方記入した事業所

この一方で，PLCアプリケーション・ソフトウェアの開発は，主に機械装置又は盤の製造業者が担当してきたが，使用者自身又は関連するソフトハウスが担当するケースも増加してきており，PLCアプリケーション・ソフトウェアの開発を取り巻く環境も変化を見せている（**図2**）。

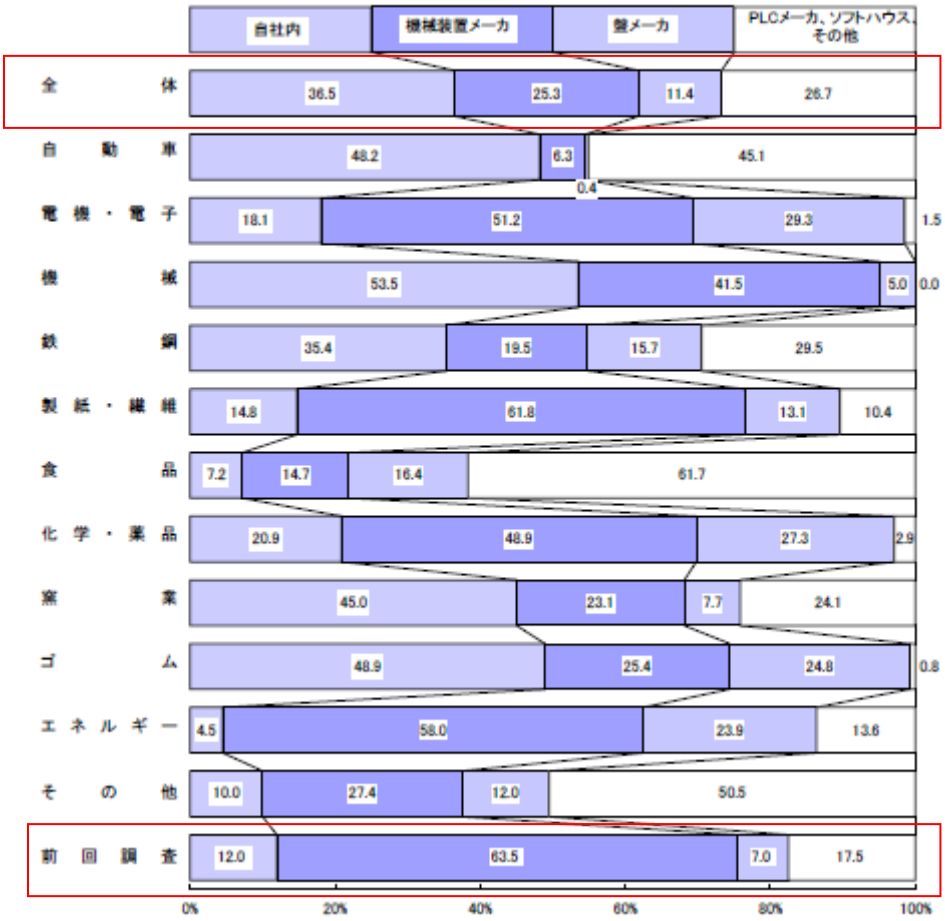


図2—PLCのアプリケーション・ソフトウェアの開発部門

こうした変化の状況は，使用者がPLCを選定するポイントとしても反映されており，かつてはあまり注目されなかったプログラミングの容易性（25.5 %）及び保守の容易性（33.7 %）が上位を占めるようになってきた。さらに，機械メーカを中心にPLC及びPLC搭載製品の輸出比率は年々増加しており，国際的なマーケットでビジネス展開するためには，地域を越え効率よく開発保守を可能とするための手段が重要になってきた。

この手段の一つが，プログラムの部品化であり，それを具現化するのがIEC 61131-3（JIS B 3503）の採用である。この規格は，世界中のエンジニアが共通の言語でコミュニケーションでき，さらに，プログラムの部品化によって効率的にプログラムを生産できるよう規定されたPLCのための国際標準プログラミング言語である。

しかしながら，日本国内では，未だこのプログラミング言語を用いた設計手法の普及が十分ではない。また，従来のラダープログラミング手法の延長線上ではソフトウェアの部品化による再利用が十分には進んでいない。その結果，製品出荷規模の増加及びシステム規模の増加が直接，設計工数の増大に繋がり，最前線のエンジニアの負担が増大していることが推定される。

これらは，必ずしもすべてが設計手法に起因する話ではないが，日本国内ではこうした課題を解決するための新たな設計手法の導入及び確立が切望されている。

この資料では，具体的な適用事例を用いて，部品化によるソフトウェアの再利用の手法及び効果を明示する。そうした手法の普及によって，ソフトウェア開発の生産性及び保守性が向上して，エンジニアの負

担が軽減する。その結果として，製品の出荷品質の向上及び市場要求への迅速な対応をもたらすことを目的としている。

※IEC 61131-3（JIS B 3503）とは

IEC 61131は，プログラマブルコントローラ（PLC）の国際標準規格として国際電気標準会議(IEC：International Electrotechnical Commission）が規定したものである。この規格は，七つのPartに分かれており，この中のPart3（IEC 61131-3）でPLCアプリケーションの標準化を目的としたプログラミング言語を規定している。JIS B 3503は，このIEC 61131-3を日本工業規格として規定したもので，規定内容としては，IEC 61131-3と同じものである。また，IEC 61131-3の推進団体であるPLCopen（日本ではPLCopen Japan）では，活動の一環として，PLCアプリケーションのマルチベンダ対応を推進している。

2　生産設備ユーザの課題

市場で競争力のある商品を製造していくうえで重要な“生産設備の構築及び維持・改善”に関わる課題とその対策を，S/W開発効率化の視点を中心にまとめる(図3)。

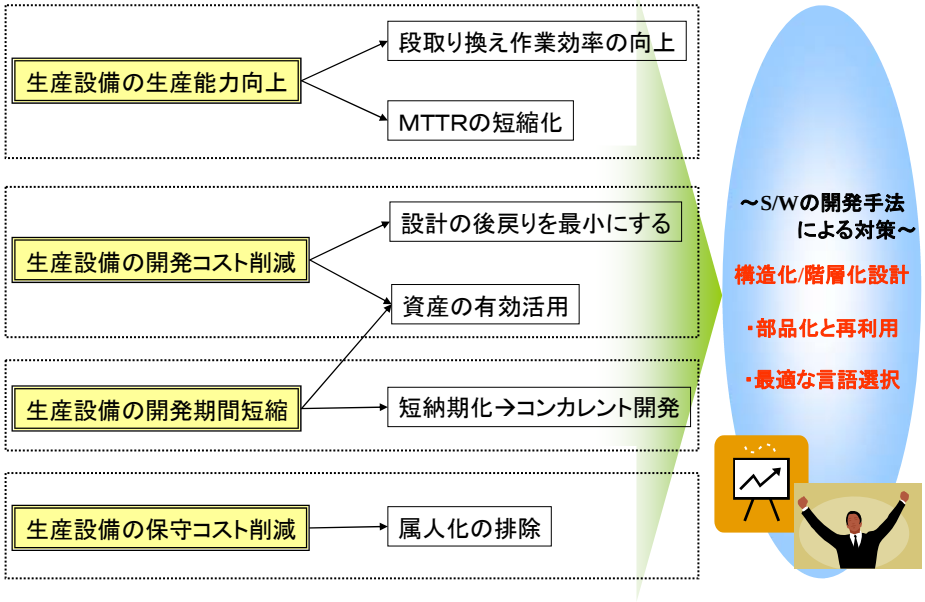


図3—生産設備ユーザの課題

2.1　生産設備の生産能力向上

2.1.1　課題

製品価格の競争力及び製品安定供給力を確保するために，生産設備の生産能力向上により生産効率を上げる。

2.1.2　対策

a)　タクトタイムの短縮

- H/W(PLC)の高速化による生産整備の性能向上を図る。

b)　設備の稼働率向上

- S/Wの構造化設計及び部品化によって，パラメータの変更を容易にし，段取換え作業の効率の向上を図る。
- S/Wの最適な言語選択による可読性向上，構造化設計及び部品化による停止要因などの各種情報収集の容易性を高め，MTTRの短縮化を図る。

2.2　生産設備の開発コスト削減

2.2.1　課題

生産製品の多様化及び高度化に対応し，製造コストの削減による製品価格の競争力を確保するために，生産設備の高度化・複雑化への対処及び開発コストの削減を両立する。

2.2.2　対策

a)　H/W開発コスト削減

- 複数社から購買，専用機から汎用機へシフトする。

b)　S/W開発コスト削減

- 外注化から内製化へシフトする。

- デザインレビュー及び開発プロセス規定の適用によって，設計の後戻りの最小化を図る。
- 構造化設計による仕様すり合せ範囲の明確化及び最小化を図り，複数人による同時作業の効率化及び設計の後戻りの最小化を図る。
- 構造化設計によるプログラムの部品化及び再利用によって，設計資産の有効活用を図る。

2.3 生産設備の開発期間短縮

2.3.1 課題

市場ニーズの多様化及び製品ライフサイクルの短期化に対応し，製品をタイムリに市場投入したい。

2.3.2 対策

a) 生産設備の新設・改造・拡張入替えサイクルの短期化

b) 生産設備の複雑化及び大形化並びに短納期化の両立

- コンカレント開発による工期短縮を図る。
- 開発対象システムのサブシステムへの分割によって，並行作業性の確保を図る。
- 構造化分析に基づく構造化設計によって，結合時の手戻りリスクの評価・低減が可能となり，開発期間の予想外の長期化を防止する。

2.4 生産設備の保守コスト削減

2.4.1 課題

設備を止められない，作業時間が限定されるなどの制限の多い条件下でも，作業時間の最小化及び保守コストを削減したい。

2.4.2 対策

a) プログラム理解の容易性確保及び技術継承

b) 機能拡張及び改変の容易性確保

- ドキュメントの充実及びプログラムの部品化によって，可読性を向上し，巻物的プログラムの排除及び属人化の排除並びに技術継承の容易化を図る。
- 構造化及び階層化したプログラム構造の導入，制御内容に最適な言語の選択，プログラム自身のドキュメント化による可読性の向上並びに解析作業の省力化を図る。
- プログラム部品の標準化(品質保証済)による再利用の促進によって，品質確保及び保守コストのミニマム化を図る。

3　効率的なPLCアプリケーション開発手順

アプリケーションの開発の手順は，図4による。

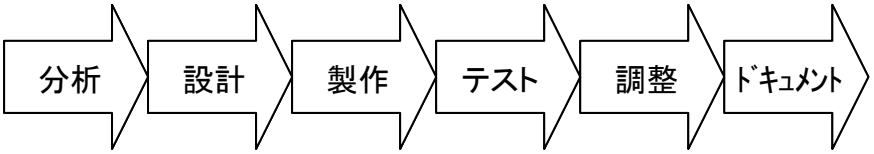


図4—PLCアプリケーション開発手順

“分析”では，システムの動作を分析し，機能単位を抽出する。“設計”では，この分析結果に基いた機能単位をファンクションブロック化するための設計をし，“製作”ではプログラミングによって，ファンクションブロック及び運転方案プログラムを作成する。“テスト”では，ファンクションブロック，機能モジュール単体などのデバッグから，全体を組み合わせたデバッグを実施する。“調整”では，制御対象に合わせたパラメータの調整を実施し，各種データを運用に最適な値に設定し，データテーブルを保存する。設計の最後には，保守及び拡張に備え，ドキュメントを出力します。

3.1　アプリケーションモデル

この章では図5に示す搬送システムの事例を用い，分析～保守の設計手順を解説する。

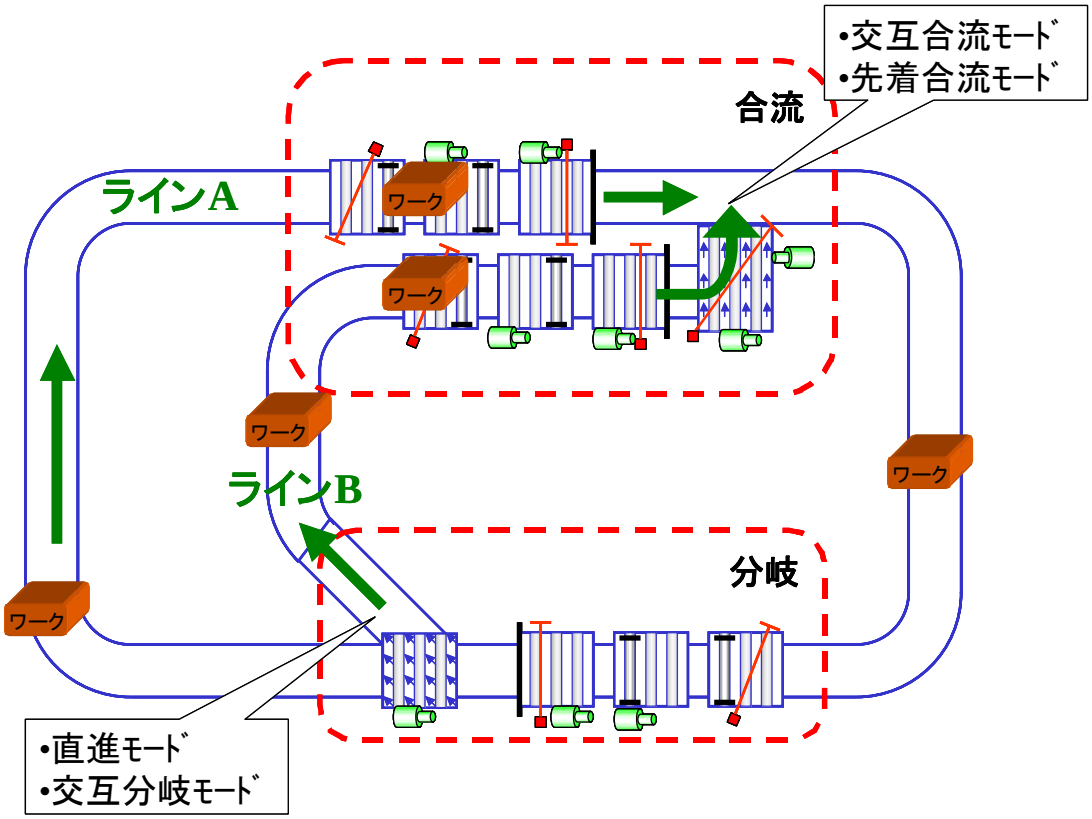


図5—搬送システムの事例

3.1.1　動作仕様

a)　搬送ラインA上(外周)をワークが流れる。

b)　搬送ラインA上の分岐ユニットにワークが到達すると，ワークは運転指示によって，ラインAをそのまま流れるか，ラインBに分岐する(表2)。

表2—搬送ラインAの動作

運転指示	動作
直進モード	ワークは，常にラインAを流れる。
交互分岐モード	ワークは，ラインA及びラインBを交互に分岐して流れる。

c)　搬送ラインB上の合流ユニットにワークが到達すると，ワークは運転指示によって，ラインAに合流する(表3)。

表3—搬送ラインBの動作

運転指示	動作
交互合流モード	常に，ラインAのワーク及びラインBのワークが交互になるように合流する。
先着合流モード	ラインAのワーク又はラインBのワークのいずれかが，先着順に合流する。

d)　ワークの渋滞によって，搬送ラインが滞った場合は，搬送ラインを停止する。

e)　後に，ラインの変更又は拡張が予想される。

3.2　分析

分析は，要求仕様の整理，機能抽出及び構造化の順に実施する。分析の詳細は，図6による。

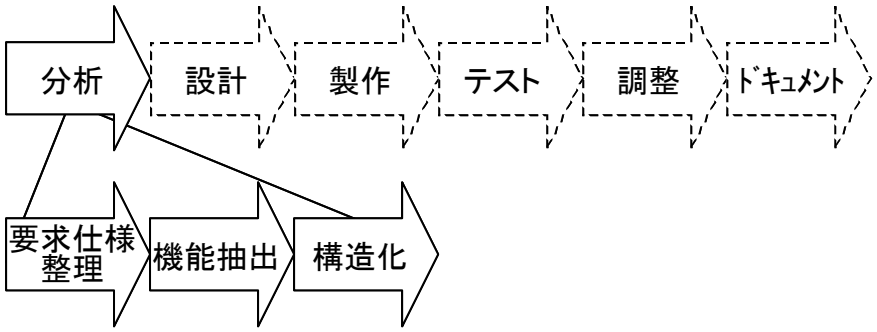


図6—分析の詳細手順

3.2.1　分析のポイント

“機能抽出・細分化”→“グループ化”→“階層化”→“構造化”のアプローチで，常に，機能単位のブロックを基本単位としてシステムの構築を考えることによって，システムごとのカスタム的な構築から，共有，再利用を中心とした高効率なシステム構築に切替え可能である。

このシステムを分析するうえでは，図7に示すようなプログラムの階層化構造をあらかじめイメージしている。機能最小単位のブロックを，基本部FB層とし，その上に小機能をまとめた大機能で構成する応用部FB層，最上位にシステム全体の動作を制御する，運転方案プログラム層で構成される。このイメージを意識しながら，システムの機能抽出～構造化分析を実施すると，機能の整理がしやすく，実際のプログラムとの対応関係も取りやすくなる。

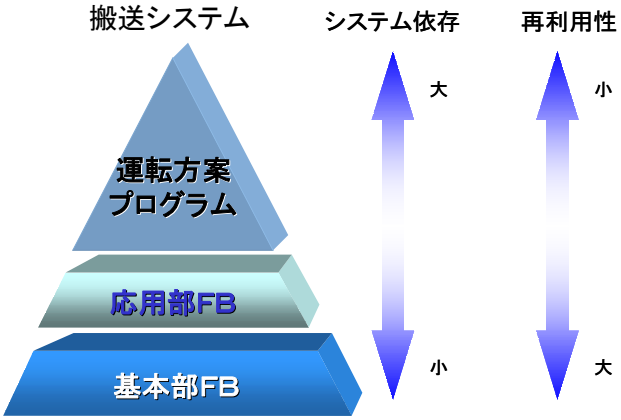


図7—S/Wの階層構造

さらに，機械ユニットと一体でFBが管理及び構築可能な場合，その効果はさらに高くなる。システムの拡張又は変更に備えて，H/W及びS/Wが対となったモジュールで構成されるシステムが望ましい姿である。

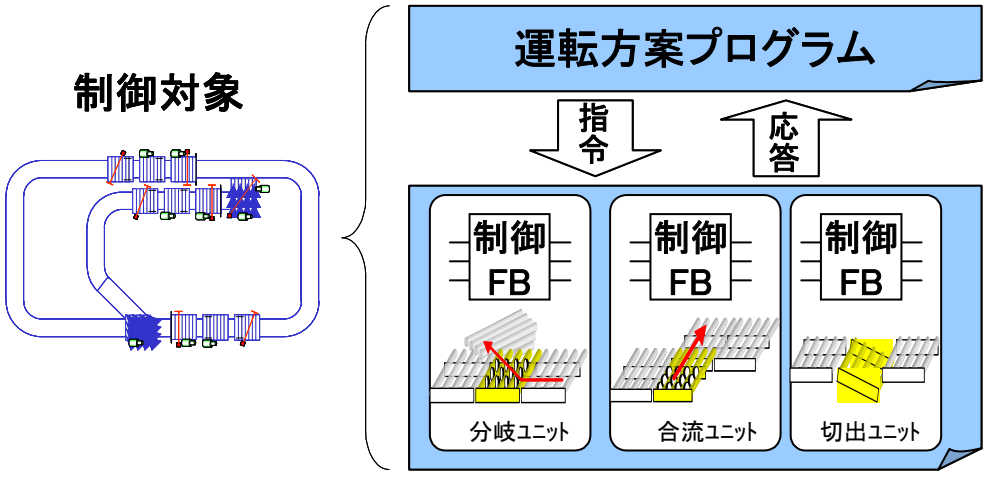


図8—運転方案プログラムと制御FB，制御FBと機械ユニットの関係

この搬送システムの分析を進めるうえでは，制御プログラムの構成を図8のようにイメージするとよい。対象設備又は機械ごとに異なる運転方案部と個々の機械ユニットを制御するFBに役割を分離する方向でアプローチすると，再利用性及びメンテナンス性の向上が期待できます。

3.2.2　要求仕様整理

システムの動作仕様が不明確ではアプリケーションを開発できない。仮に，不明確のままで開発を進めた場合，開発の終盤で手戻りが発生したり，完成品の機能又は品質が使用者にフィットしないなどの，膨大なロスの発生要因となることはいうまでもない。

したがって，システムの動作仕様を決定するために，要求仕様を整理し，曖昧さをなくし動作仕様を明

確にすることが大切である(この資料では，既に要求仕様が明確である前提で説明を進める。)。このシステムの動作仕様は，3.1による。

3.2.3　機能要素の抽出

まず，搬送ユニットの動作及び機能に着目し，機能要素単位を整理する(図9)。

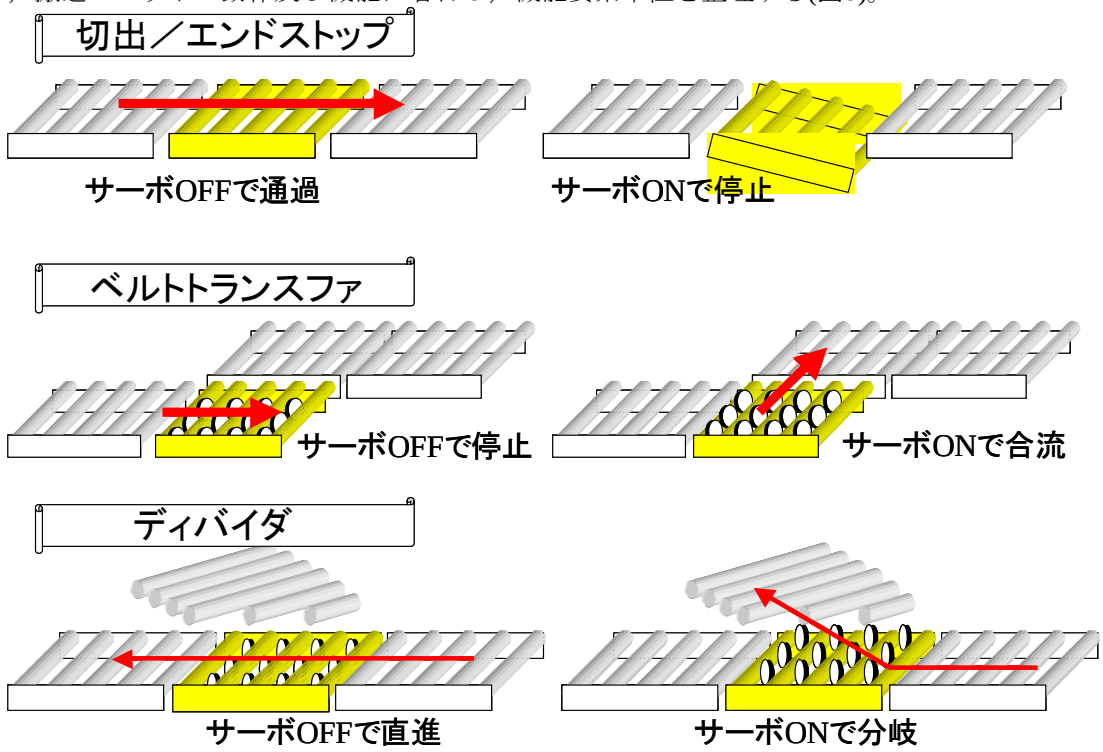


図9—搬送ユニット

この搬送システムは，主に，3パターンの機能を果たす搬送ユニットによって構成される。機能要素の抽出分析には，様々な方法が考えられるが，アプローチ例は，次による。この例ではa)のアプローチを採用している。

a)　加工物(ワークの流れ)を中心に，機械・設備の機構部の動作を分析し，パターン化し，機能要素ブロックを抽出する。

b)　動作フローから制御フローを検討し，制御処理の中から機能ブロックを抽出する。

c)　データ及び信号の流れに着目し，機能ブロックを抽出する。

d)　工程に着目し，機能ブロックを抽出する。

e)　繰返し用いることができそうな部分及び用いることができない部分に着目し，機能ブロックを抽出する。

次に，“合流”及び“分岐”の機能を，抽出した搬送ユニットの機能要素単位を用い，細分化する(トップダウンアプローチ)。逆の見方をすると，抽出した搬送ユニットの機能要素単位を，“合流”及び“分岐”の機能にグルーピングする(ボトムアップアプローチ)。

さらに，ワークの渋滞を監視するために，ワークが停滞する可能性がある各ポイントに，満杯検出機能が必要であることがわかる(図10)。

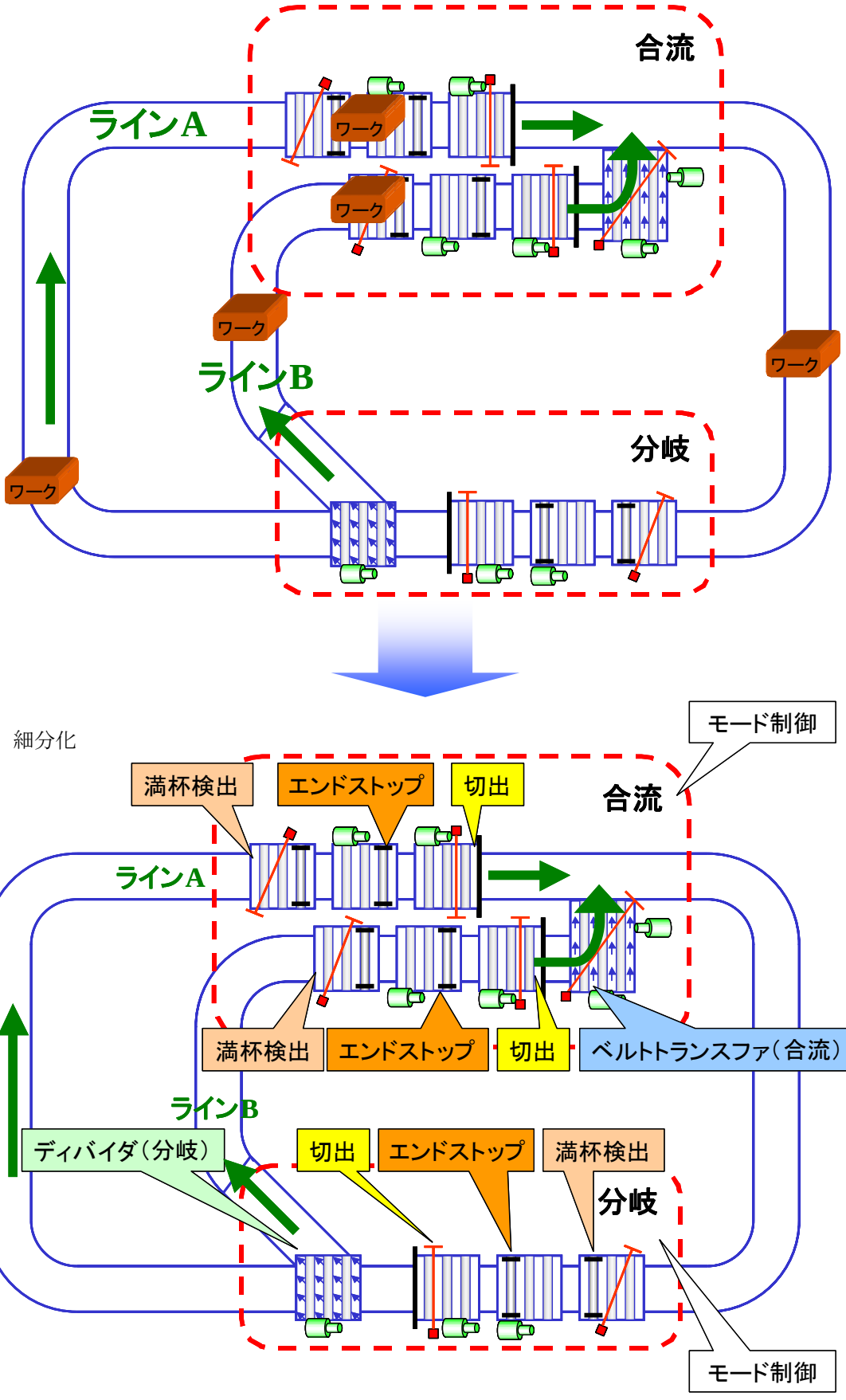


図10—搬送システムの機能分解

3.2.4　構造化

各搬送ユニットに指示を送り，搬送システム全体の動作を指示する機能を“運転方案”部とし，3.2.3で分割した“搬送ユニットの個々の機能”部との関連(相関)を分析すると，図11に示す階層構造となる。別の機能間でも，機能が重複しており，その機能ブロックは共通化できることがわかる。

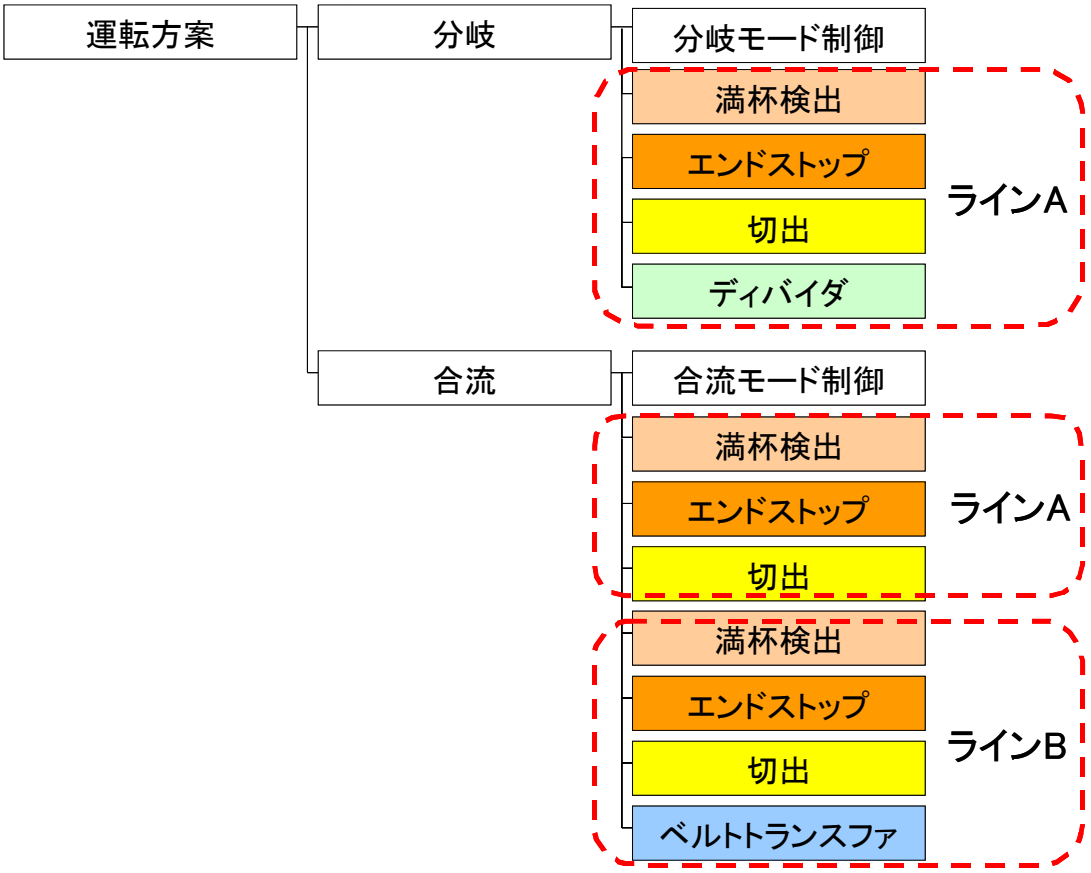


図11―搬送システムの機能階層

物理的な機器(機械)の機能，論理的な工程(プロセス)などの切り口で分析すると，その構造は階層構造をとる場合が多く，大規模になればなるほどその傾向は強くなる。階層構造の分析を進めて共通機能を抽出すると開発量を最適化することができる。さらに，その部分に既に存在する機能部品を用いることができる場合，開発量を最小限にすることが可能である。

3.3　設計

ここでの設計とは，3.2での分析結果を元に，搬送システムの機能要素をPLCアプリケーションのS/W要素，すなわち，FBへ当てはめ，少しずつ仕上げていくことを意味する。

なお，FBの概要及び特長については，箇条5を参照されたい。

3.2の分析結果によって，図12の階層構造が得られている。

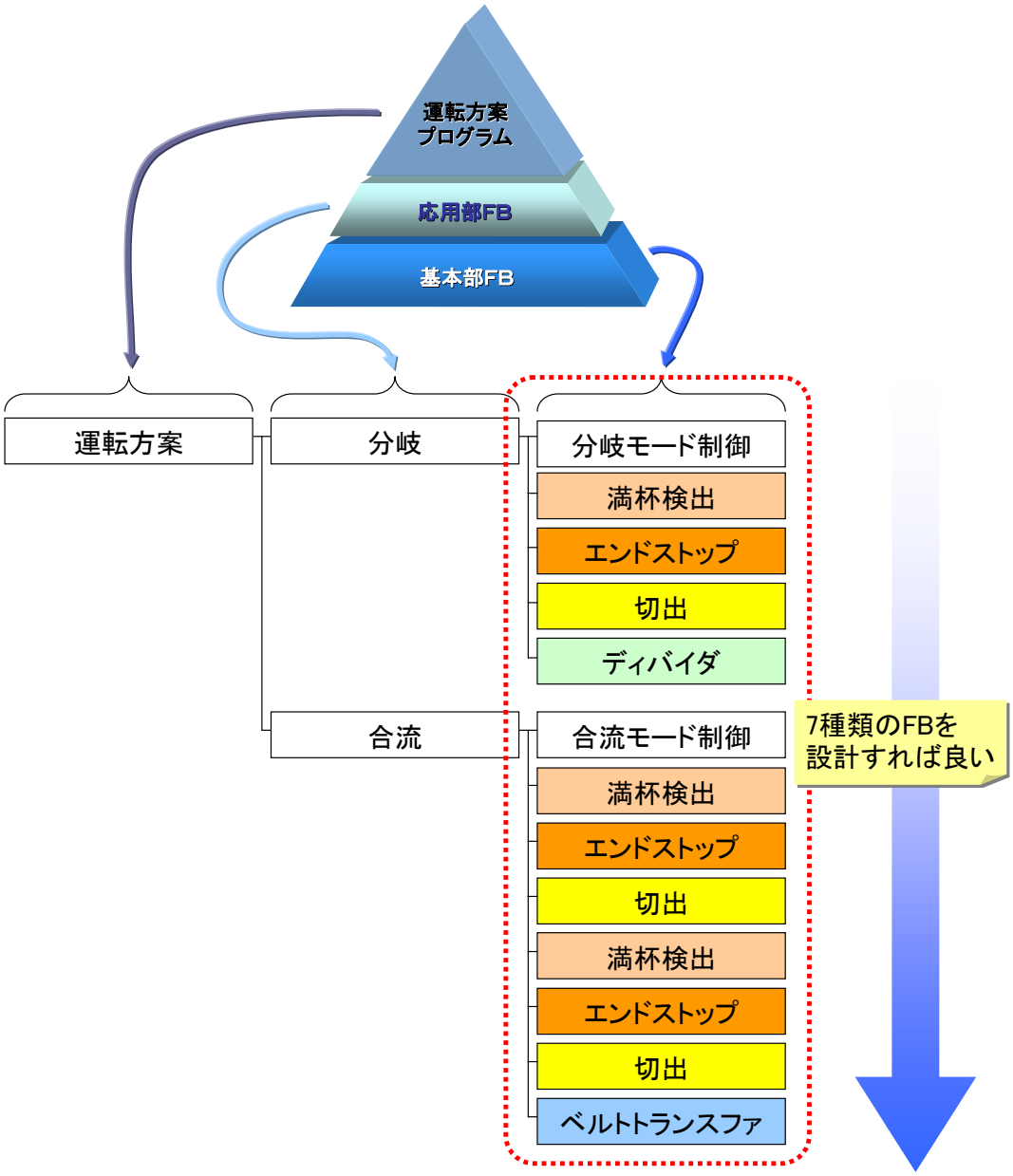


図12—搬送システムの機能階層と基本部

まずは，比較的FB化しやすい最下層の基本部(破線囲みの部分)から設計を始める。この部分は，機能要素が単純な動作仕様にまで明確になっている場合が多く，また，一つの機能がそのまま一つのFBに対応させやすい場合が多い。その後で，これら基本部の小機能を組み合わせて，より大きな機能から構成される応用部の設計を行う。

基本部FBには，例えば“満杯検出”機能が三つある。このとき，“満杯検出”機能を一つのFBとして設計・製作(プログラミング)すれば，残りの“満杯検出”はその流用で済む。そのため，破線囲みの部分には13の機能があるが，実際には，7種類のFB(満杯検出FB，エンドストップFB，切出FB，ディバイダFB，ベルトトランスファFB，分岐モード制御FB，合流モードFB)の設計・製作(プログラミング)で済むことになる。

3.3.1　基本部FBの設計

基本部FBは，図13に示す搬送ユニットのFB及び分岐・合流モード制御のFBの二つに大きく分けられる。

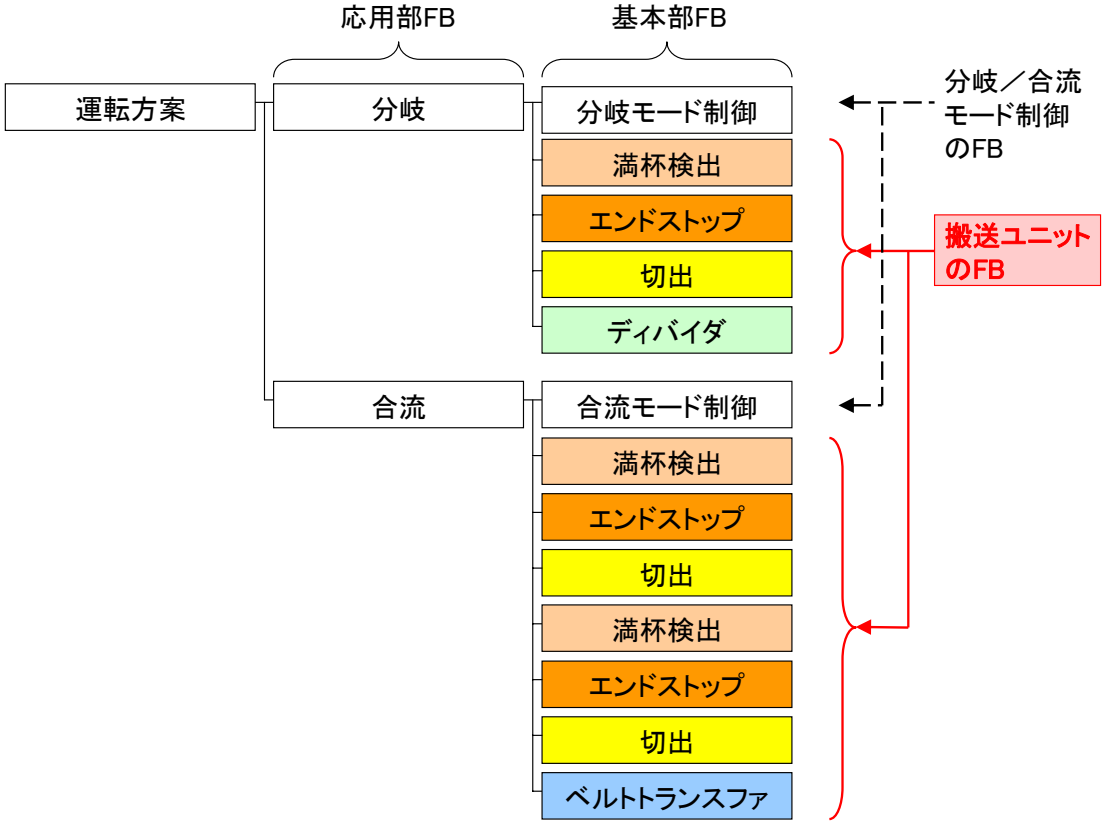
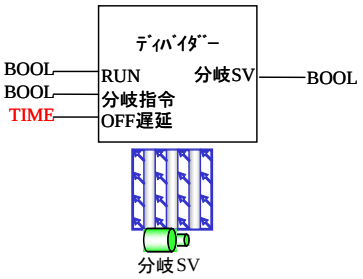
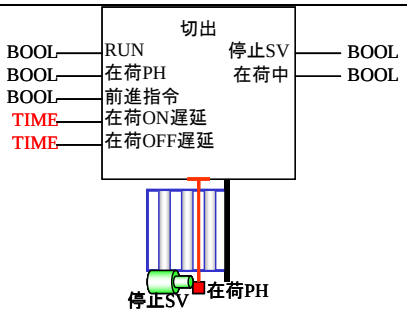
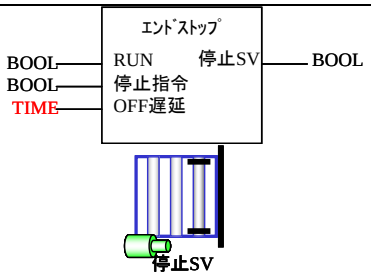
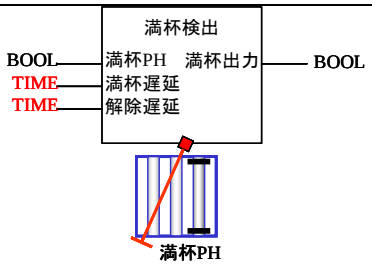
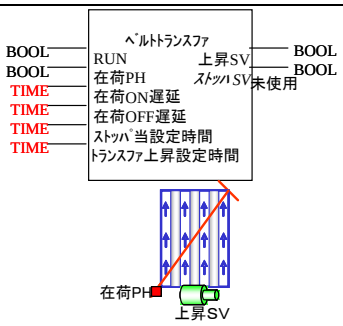


図13―基本部FB(S/W階層の最下層部分)

a)　搬送ユニットのFB設計

機能の階層構造の最下層である基本部には，搬送ユニット(検出器を含む)という機器の制御を担当する部分がある。まずは，これらの制御を入力，制御内容及び出力という観点で仕様を明確にする。次に，制御の入力及び出力をFBの入力ピン及び出力ピンというように割り当てていく。この作業の結果，基本部の制御FBは，表4のようになる。

表4—搬送ユニットFBの制御仕様・入出力仕様

搬送ユニット名 （基本FB名）	搬送ユニット制御仕様	搬送ユニットFBの入出力仕様
ディバイダ	分岐指令によって分岐SVをON（荷を分岐）。 分岐指令解除後，OFF遅延時間後，分岐SVをOFF（荷を直進）。 RUN信号の解除時は分岐SVを強制OFF。	
切出	在荷PHによって荷を検出。 在荷ON遅延時間及び在荷OFF遅延時間の監視によって，ストッパ到達・到達解除を判定。 荷のストッパ到達によって，停止SVをON（荷の進行停止）。 荷の到達解除又は前進指令によって，停止SVをOFF（荷を進行）。 RUN信号の解除時は停止SVを強制OFF。	
エンドストップ	停止指令によって，停止SVをON（荷の進行停止）。 停止指令の解除からOFF遅延時間後に停止SVをOFF（荷を進行）。 RUN信号の解除時は停止SVを強制OFF。	
満杯検出	満杯PHによって荷の到着を検出後，満杯遅延時間の監視によって，満杯を判定。 満杯PHによって荷の未到着を検出後，解除遅延時間の監視によって，満杯解除を判定。	
ベルト トランスファ	在荷PHによって荷を検出。 在荷ON遅延時間及び在荷OFF遅延時間の監視によって，ストッパ到達・到達解除を判定。 荷のストッパ到達にて，上昇設定時間だけトランスファ上昇SVをON（合流）。 時間監視によって，合流完了を判定。 トランスファを下降。 RUN信号の解除時は上昇SVを強制OFF。	

切出のFB例は，**図14**による。

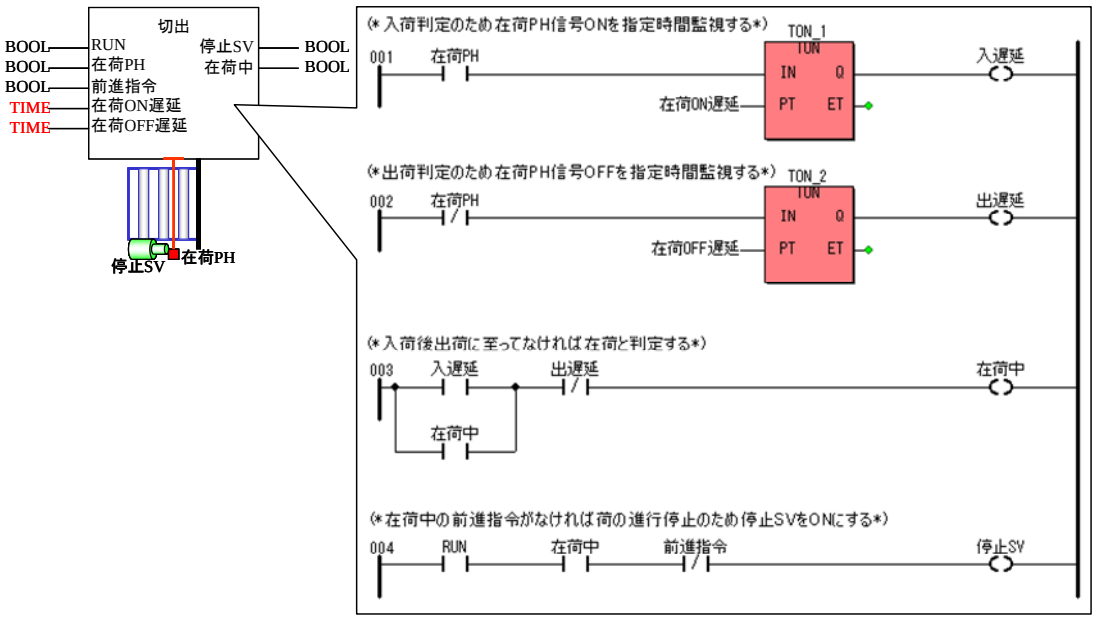


図14—切出のFB例

b)　分岐・合流モード制御のFB設計

前述a)の搬送ユニットのFBを単に集めても，“分岐”及び“合流”という機能を実現することはできない。各搬送ユニットのFBに適切な指示を与えたり，逆に搬送ユニットからの応答又は各種検出結果を受けたりすることで，動作仕様にある“分岐モード”及び“合流モード”の制御(破線囲みの部分)を担当する機能が必要になる。

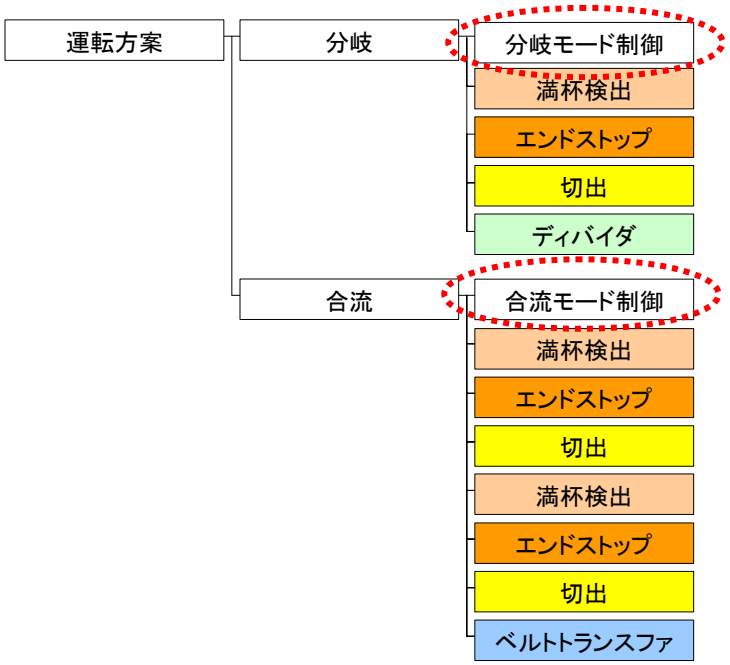
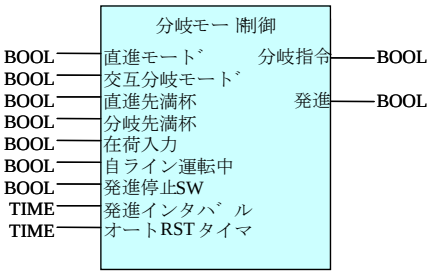
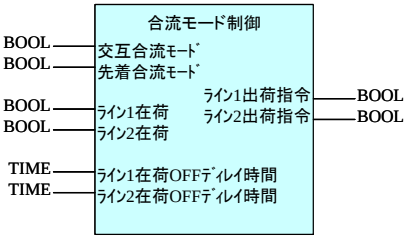


図15—分岐／合流モード制御のFB

分岐モード制御及び合流モード制御についても，搬送ユニットと同様に，まずはこれらの制御を入力，制御内容及び出力という観点で仕様を明確にする。次に，制御の入力及び出力をFBの入力ピン及び出力ピンというように割り当てる。この作業の結果，分岐・合流モード制御FBは，**表5**のようになる。

表5—分岐／合流モード制御FBの制御仕様・入出力仕様

機能名 （基本FB名）	モード制御仕様	モード制御FBの入出力仕様
分岐モード制御	自ライン運転中での切出ユニットからの在荷を確認する。 直進モードがONのときには荷の分岐指令はOFF, 逆に直進モードがOFFのときには荷の分岐指令をONにする。交互分岐モードがONのときには荷の発進ごとに分岐指令を交互にON/OFFする。 直進先満杯又は分岐先満杯がONのときには荷の発進は行わない。また，発進停止SWがONの場合にも荷の発進は行わない。 荷の発進後，発進インターバル時間は次の荷の発進は行わない。 荷の分岐指令がオートリセットタイマ時間が続いた場合には，強制的に分岐指令をOFFにする。	
合流モード制御	※省略	

3.3.2　応用部FBの設計

次に，機能の階層構造の中間層に当たる応用部の設計を行う。この応用部は，3.3.1において設計した基本部の小機能（搬送ユニット制御，分岐・合流モード制御）をグループ化して，運転方案を構成する要素である“分岐”機能及び“合流”機能にする。

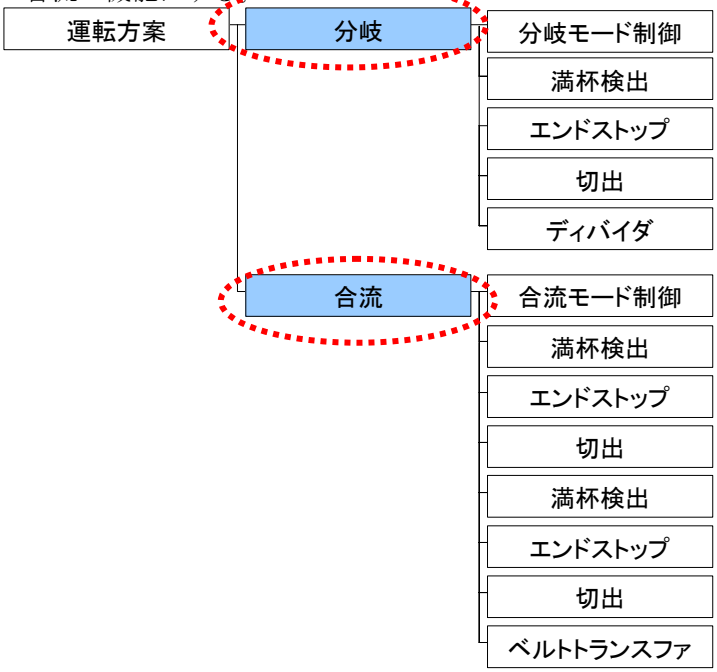


図16―応用部FB(S/W階層の中間層部分)

中間階層に位置する応用部のFBのインタフェースを設計する場合，その内部の基本部FBで用いる外部I/O信号は，応用部FBの入力ピン又は出力ピンを介して応用部FBの外側から受け渡すとよい。こうすることで，H/W構成の変更に伴って外部I/O信号の物理アドレス（I/Oアドレス）が変更になった場合でも，この応用部FB及び基本部FB両方の中身を変更する必要がなく，FBの流用性が高まる。

例えば，“分岐”機能の内部で用いる外部I/O信号には，表6に示すものがある。

表6―分岐FBの外部I/O信号

応用部FB	基本部FB	外部I/O信号	
		入力信号	出力信号
分岐	分岐モード制御	直進先の満杯PH検出 分岐先の満杯PH検出 発進停止SW信号	―
	満杯検出	満杯PH検出	―
	エンドストップ	―	エンドストップSV（停止SV）指令
	切出	在荷PH検出	切出SV（停止SV）指令
	ディバイダ	―	ディバイダSV（分岐SV）指令

これらの入力信号及び出力信号は，そのまま“分岐”機能FBの入力ピン及び出力ピンとして用意する。さらに，運転指示信号(直進モード・交互分岐モード)及びライン運転信号(運転信号がOFFの場合は各SV強制OFF)の入力ピンを合わせた結果，“分岐”機能FBは，図17に示すピン配置とすることができる。

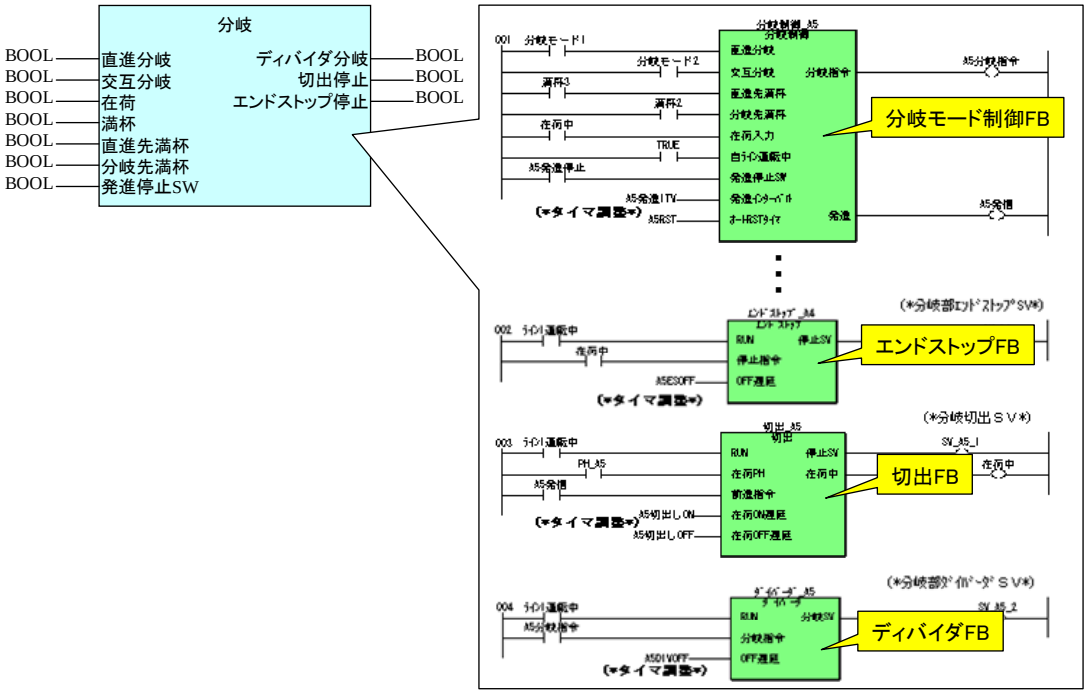


図17—分岐のFB例

※“合流”機能FBについては，説明を省略する。

3.3.3　運転方案プログラムの設計

S/W階層の最上部に位置する“運転方案”プログラムでは，中間層の応用部FBを組み合わせて，システムの動作を完成させるとともに，外部I/O信号及び指示系との信号を結びつけます。

例えば，分岐部分を実現するには応用部の“分岐”FBに各種センサ及びスイッチからの入力信号(①～⑤)及び搬送機器のSV駆動(⑥～⑧)への出力信号を結び付ける。

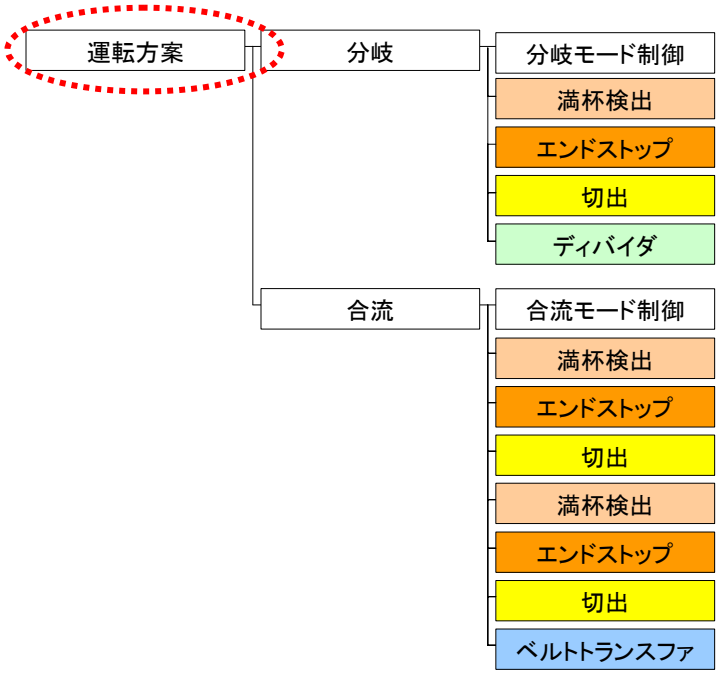


図18―運転方案プログラム(S/W階層の最上層部分)

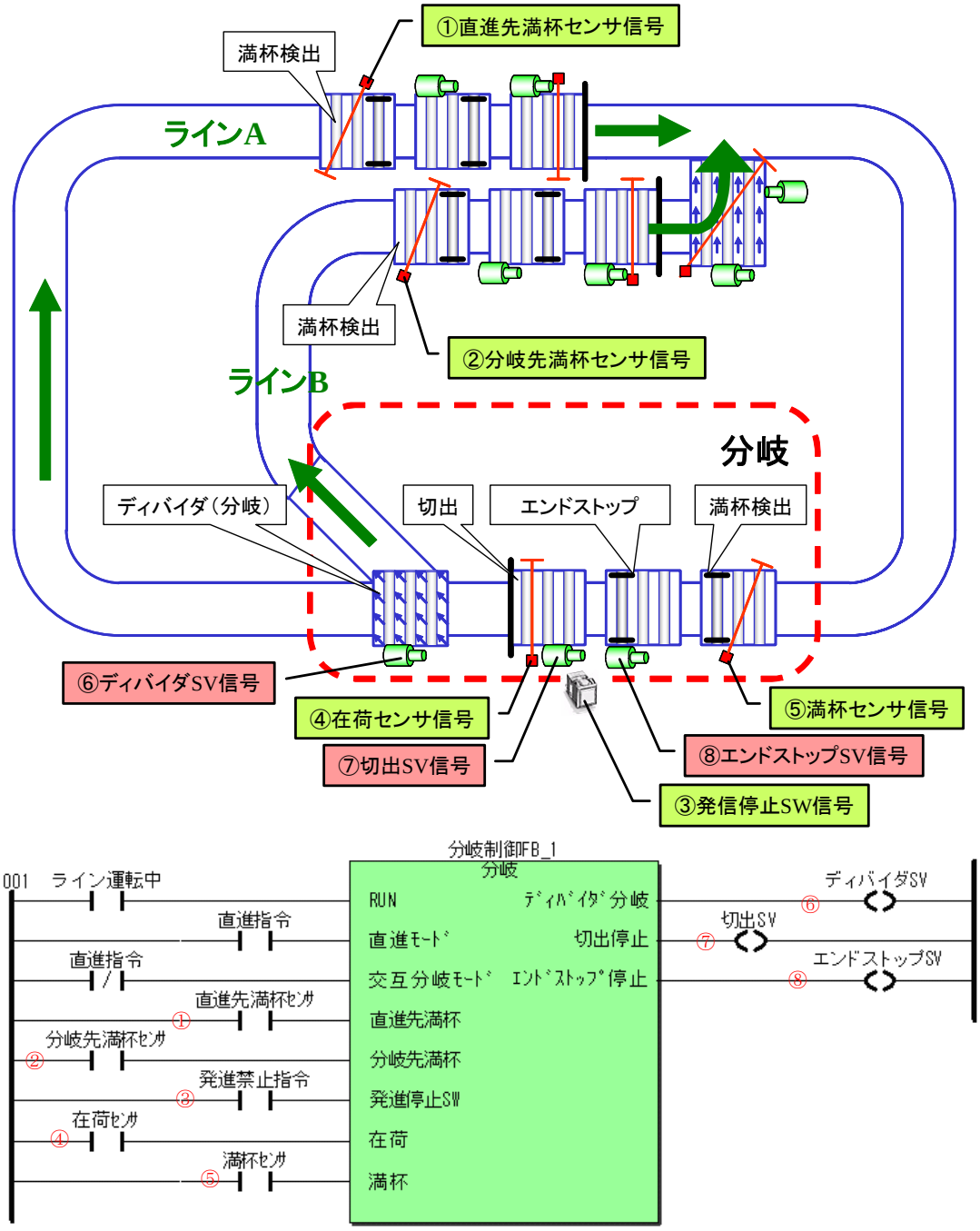


図19—分岐部分へのI/O信号の接続例

同様に，合流部分を実現するには，応用部の“合流”FBに各種センサ及びスイッチからの入力信号(⑨～⑪)及び搬送機器のSV駆動(⑫～⑯)への出力信号を結び付ける。

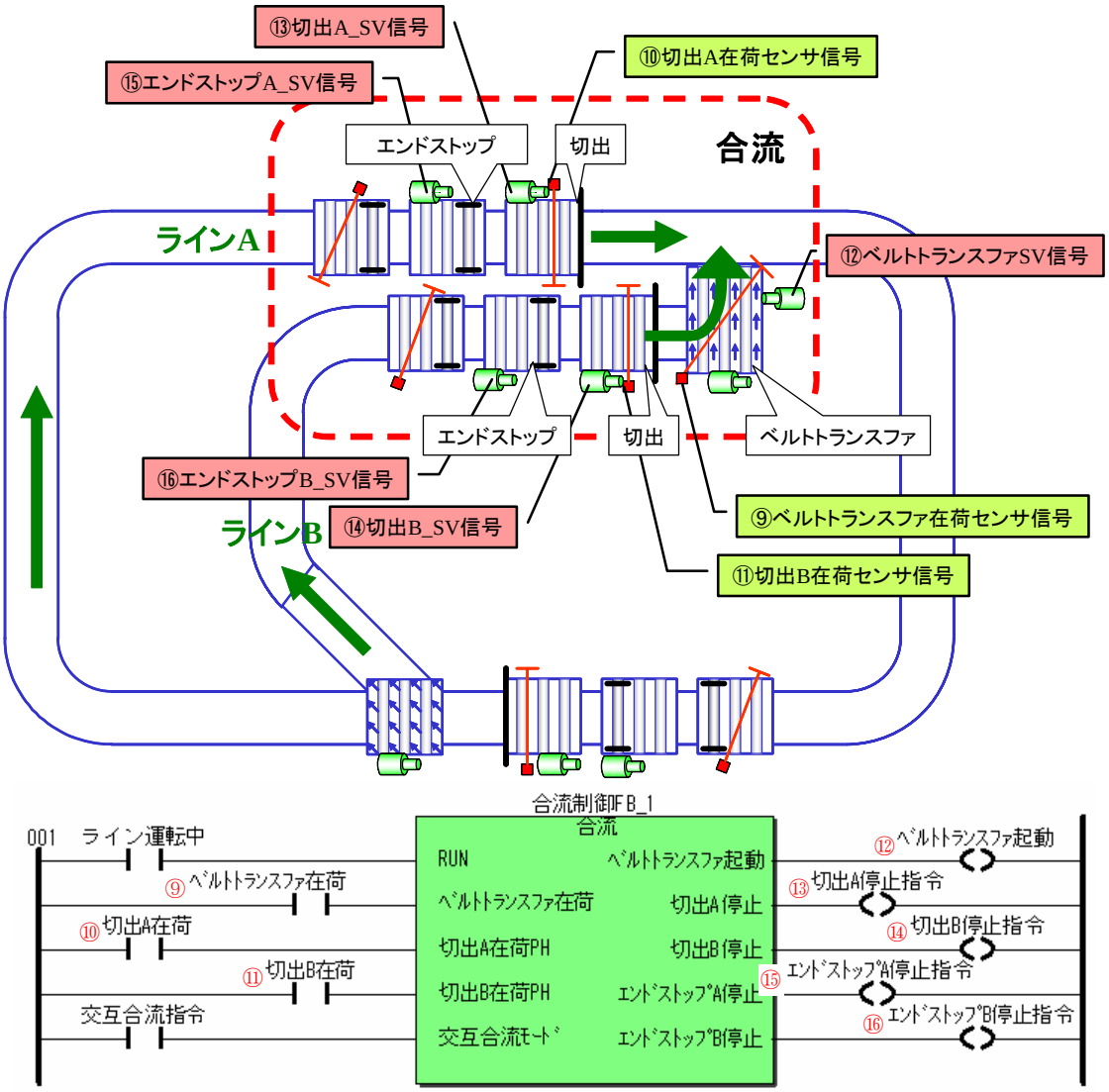


図20—合流部分へのI/O信号の接続例

最後に，応用部の“分岐”FBと“合流”FBとを結び付ける処理を記述する。

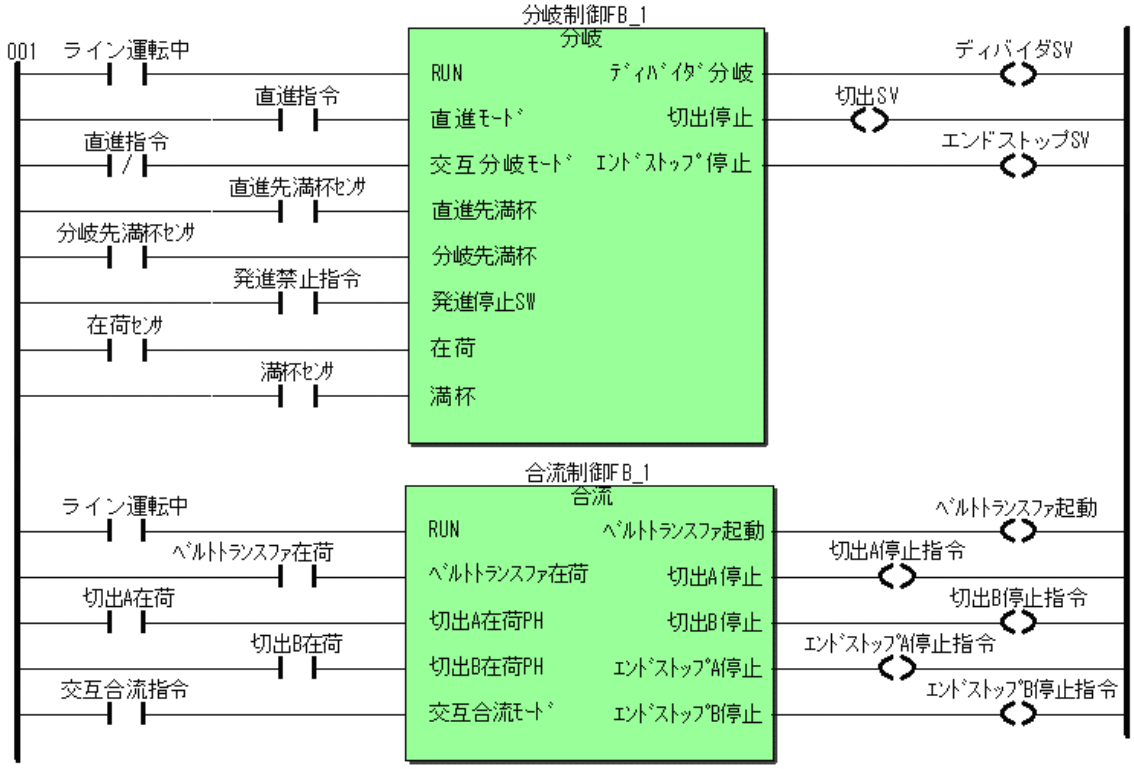


図21—運転方案のプログラム例

3.3.4　階層設計のポイント（機能の階層構造を適切にFBの階層構造に反映する）

FB適用の第一ステップは，前述の基本部・応用部のFB設計のように，機能の階層構造をそのままFBの階層構造に反映してみることである。できあがったS/Wの構造が機能，工程又はH/Wの階層構造に沿っている場合には，そのS/Wは非常に自然で理解しやすくなる（可読性の向上）。また，システムの一部変更があった場合，通常は特定の機能，工程又はH/Wの単位で変更されることが多く，S/Wの変更箇所を容易に特定でき，その変更の範囲を比較的局所化できるようになる（保守性の向上）。ただし，必ずしも機能の階層構造がそのままFBの階層構造にいつも合致する訳ではない。機能の階層構造を起点に柔軟さをもってFBの階層構造にアレンジすることも必要である。また，分析が不十分で，機能構造がきちんと把握できていない場合又は矛盾を含んでいる場合には，その機能構造をFBの階層構造に反映することが困難になる場合がある。このような場合，もう一度分析に立ち返るとよい。

3.3.5　その他の設計ポイント

ここでは，機能の構造からFBを用いたS/W構造を設計する場合のポイントを列挙する。

a)　**共通要素を部品化する**　3.2の分析の過程において機能又はH/Wの共通部分を洗い出し，それを一つのFBとしてまとめた。共通要素を一つのFBにまとめることで，製作規模（プログラムステップ数）及び単体テスト工数を削減することが見込まれる。共通要素の洗出しには，次の方法がある。

- システムの複数個所で用いられている機器がある場合，これらを制御する処理は共通要素とすることができないか検討する。
- システムの複数個所で同じプロセスを行う工程（又はサブ工程）がある場合，これらを制御する処理は共通要素とすることができないか検討する。
- 同じ計算式又はアルゴリズムを何度も繰返し演算する処理は共通要素とすることができないか検

討する。

さらに，上記の共通要素となる処理を抽出する場合，処理が全く同じではない部分があっても，それを外部から与えるパラメータによって処理内部で場合分け(機能スイッチによる処理分岐)するなどの工夫によって，共通化できる場合もある。ただし，過度の共通化のためにパラメータによる機能スイッチを増やしていくことは，そのFB内部の処理記述が複雑になり保守性を下げてしまうことがある。また，実際にFBを用いる場合に機能スイッチの組合せが複雑になってしまうこともあり，注意が必要である。このような場合には，無理に処理を共通化せず，個別の目的を果たすFBとして分けたほうがよい。

b) **メーカ提供のFBを活用する**　PLCメーカから提供されるライブラリは，品質が保証されているとともに，動作仕様，使用条件，入出力の定義などが明確に定義されているため，比較的容易にFBを用い，メリットを実感することができる。メーカ提供ライブラリにどのようなものがあるかを熟知することが，無駄な設計を抑えることにもつながる。

c) **汎用部品をライブラリ登録する**　プログラム開発の効率を向上させるために，既存資産を活用することも有効な手段の一つである。そこで，今後のS/W流用を計画し，汎用性が高い部分を集約した独立部品(FB化)とし，社内の設計者間で標準的に用いることができるS/W部品集(ライブラリ)として用意することで，将来のS/W生産性向上に寄与することができる。これは，前項の共通要素のS/W部品化において，次の観点で候補を挙げるとよい。

- システムの構成機器が将来的にも標準的に採用されるものである場合は，これらを制御する処理部分(駆動，通信，状態検出など)を標準FBとしてライブラリに含めることを検討する。
- はん(汎)用性が高い計算アルゴリズムである場合は，これを独立した標準FBとしてライブラリに含めることを検討する。

ライブラリとしてほかの設計者にも標準的に使用可能なS/W部品(以下，“標準FB”という。)とする場合には，次の点に注意するとよい。

標準FBを設計する場合の注意点は以下の通りである。

- 一つの標準FBには，多くの目的の機能をもたせすぎない(標準FBの機能が明確に保たれているほうが用いやすい。)。
- 標準FBの入力変数・出力変数は，使用時の組込みを考慮し，名称，データ形及び，並び(類似機能のFBと並びを合わせるなど)を分かりやすくする。また，入力変数・出力変数の数が多くなりすぎないように注意する。
- 標準FBでは，内部の処理でグローバル変数を参照しないようにする(外部変数は用いない。)。
- 使用時の誤解を防ぐために，その標準FBの機能及び入力変数・出力変数・外部変数を文書(プログラム上のコメントを活用してもよい。)によって明記する(3.6のドキュメンテーションを参照。)。
- 標準FBの変更履歴を残す。
- 欲張って多くの標準FBをつくらない(用いられていない標準FBばかり増えてしまうことを避ける。)。
- 標準FBの一覧(目録)を利用者の目の届くところに用意する。

これらの注意事項は，いずれも使用の側面で配慮の必要があるものである。ライブラリとして提供される標準FBは，利用されてこそ価値があるものだからである。

3.4　製作

ここでは，3.3での設計で得られたFBの定義(制御仕様及びその入出力の定義)に従って，プログラミング

していく。

3.4.1　プログラミング言語の選択

PLCのプログラミング言語として，従来から用いられてきたラダー図言語（ラダー回路）に加えて，近年，IEC 61131-3で規定された次の言語を用いることができるプログラミングツールも登場するようになった。詳細は，箇条5を参照されたい。

表7—IEC61131-3のプログラミング言語

プログラミング言語	記述形式	特長
ラダー図言語 （LD言語）	グラフィック	従来から日本国内で最も用いられている言語であり，シーケンス処理の記述に適している。
命令リスト言語 （IL言語）	テキスト	アセンブラに似たローレベル言語であり，高速に動作する小さいモジュールを記述するのに適している。
構造化テキスト言語 （ST言語）	テキスト	パソコン用プログラム言語に似たテキスト言語で，数式演算の記述又はIF...THEN文並びにFOR...DO文のような分岐制御の記述に適している。
ファンクションブロック図言語 （FBD言語）	グラフィック	FB同士のピンをつなぎ合わせて電子回路図のように記述できる言語であり，連続したデータ処理の記述に向いている。最近では，ラダー図言語と混在記述できるプログラミングツールもある。
シーケンシャルファンクションチャート言語 （SFC言語）	グラフィック	状態せん（遷）移に基づくフローチャートであり，順序制御処理の記述に向いている。

これらのプログラミング言語には，それぞれ得意，不得意があり，適材適所で用いるプログラミング言語を選択するとよい。

表8—プログラミング言語の得意・不得意

主な使用状況	LD言語	IL言語	ST言語	FBD言語	SFC言語
単純なリレーシーケンス処理	◎	×	△	△	×
数式演算処理	△	×	◎	○	×
状態せん移に基づく順序制御 （ステップシーケンス処理）	△	×	○	×	◎
連続的なアナログ信号処理	△	×	○	◎	×
複雑な情報処理	△	×	◎	△	×
プログラムメモリ制約の厳しい場合	○	◎	△	△	×
最も高速に性能を求められる場合	○	◎	○	○	×
運転方案と対応がとりやすい表現	×	×	○	○	◎
動作を視覚的に確認したい場合	○	×	×	◎	○
注記　記号の意味は，次による。 ◎：最も適している，○：適している，△：困難な場合もある，×：適さない					

3.4.2　変数の命名について

データ又は信号を受け取ったり，又は一時的に保持したりするために変数が用いられる。この変数には，個別に“変数名”をつけて用いる。変数名は，自由な名称を設定することが可能だが，その変数の用途，格納するデータ種別などによって名称の付け方を工夫することでプログラムの可読性を上げ，後々の保守において処理内容を理解する助けとなる。

大切なことは，簡単なルールでもよいので事前に用意しておくことである。特に，多人数でプログラムの作成及び保守を行う場合には重要なことである。

a) 処理の理解を助ける命名

プログラムで用いられる変数について，その変数の使用目的に沿った意味ある名前で命名されていると，プログラムを理解しやすくなる。

変数名に信号の意味を明確に示さない場合のプログラム例は，次による。

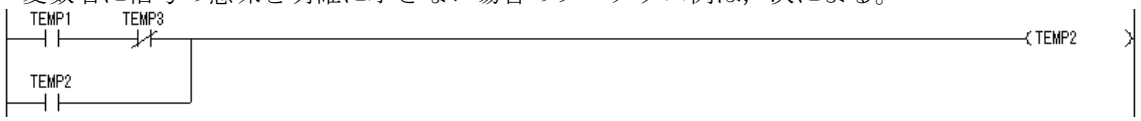


図22—信号の意味が分りにくい変数名のプログラムの例

上記と同じ処理を行うプログラムでも，変数の名称が使用目的を端的に表現することで，処理の内容及び意図が汲み取りやすくなる。

図23—信号の意味を汲み取りやすい変数名のプログラムの例

b) 仮名・漢字を用いた命名

変数名に仮名・漢字を用いると，端的で分かりやすく命名することができる。ただし，PLCシステムを海外に輸出する場合，現地のプログラミングツールが正しく表示できないこと，現地エンジニアが仮名・漢字が読めない場合には保守できないことなど，現地の環境を考慮されたい。

c) 変数命名のポイント

変数の命名方法のポイントは，次による。

- システム仕様書などの上流仕様書で用いられている用語を変数名に用いる。上流設計との一貫性が保ちやすくなる。
- 変数名を短くするための略称は，幾つか事前に整理して，部門間で共通したものを用いる。別のエンジニアが知らない無理な略称は，せっかくの意味ある命名でも理解することが困難になる。

d)　プログラム中で識別しやすい命名の例

変数の命名に特別なルールを設け，変数名からほかの変数とは区別できるようにしておくと便利な場合がある。

例えば，グローバル変数はあらゆるプログラムの場所から値の設定・取得が可能な変数である。このグローバル変数を扱う処理を変更する場合，ほかの部分にも影響を与えることがある。そのため，表9のようにグローバル変数をほかの変数と区別するためのプレフィックス(接頭辞)を用いて命名しておくと，プログラムのコード記述の中から見つけやすくなり，プログラム変更の際の注意を引くことができる。

表9—変数の命名ルールの例

分類	プレフィックスの例*	命名の目的	変数名の例
一般的なグローバル変数	“g_”から始める (Globalの頭文字)	ローカル変数と区別をつける。	g_SystemStatus (システム状態格納用)
ネットワークリンクに割り付けられた変数	“n_”から始める (Networkの頭文字)	ほかのPLCシステムと影響しあう変数であることを区別する。	n_Station2_SystemStatus (2号局用システム状態格納用)
外部入力信号	“i_”から始める (Inputの頭文字)	H/W入力信号の変数であることを区別する。	i_StartSwitch (開始スイッチ入力用)
外部出力信号	“o_”から始める (Outputの頭文字)	H/W出力信号の変数であることを区別する。	o_ErrorStatusLamp (エラー点灯出力用)
ローカル変数	上記のプレフィックスは用いない。	—	TriggerCount (トリガー内部計測値)
注* プレフィックスの表現は自由である。ほかにも，外部入力信号及び外部出力信号の表現にx_，y_，q_なども考えられる。			

3.4.3　FB及びFB変数の命名について

FBを作成する場合につけるFB名及びFBをプログラム内で用いる場合につけるFB変数名(インスタンス名とも呼ばれる。)について，十分に配慮する必要がある。これは変数の命名と同様であり，プログラムの可読性を上げ，後々の保守においても処理内容を理解する助けとなる。

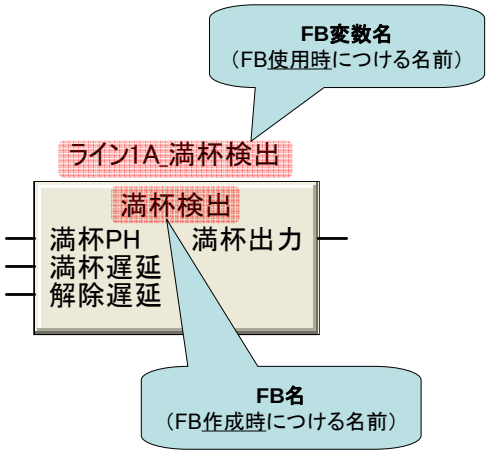


図24—FB名とFB変数名

FB作成時につけるFB名は，機能を端的に表す名称とし，FB使用時につけるFB変数名は，制御対象を区別する名称にするのがよい。

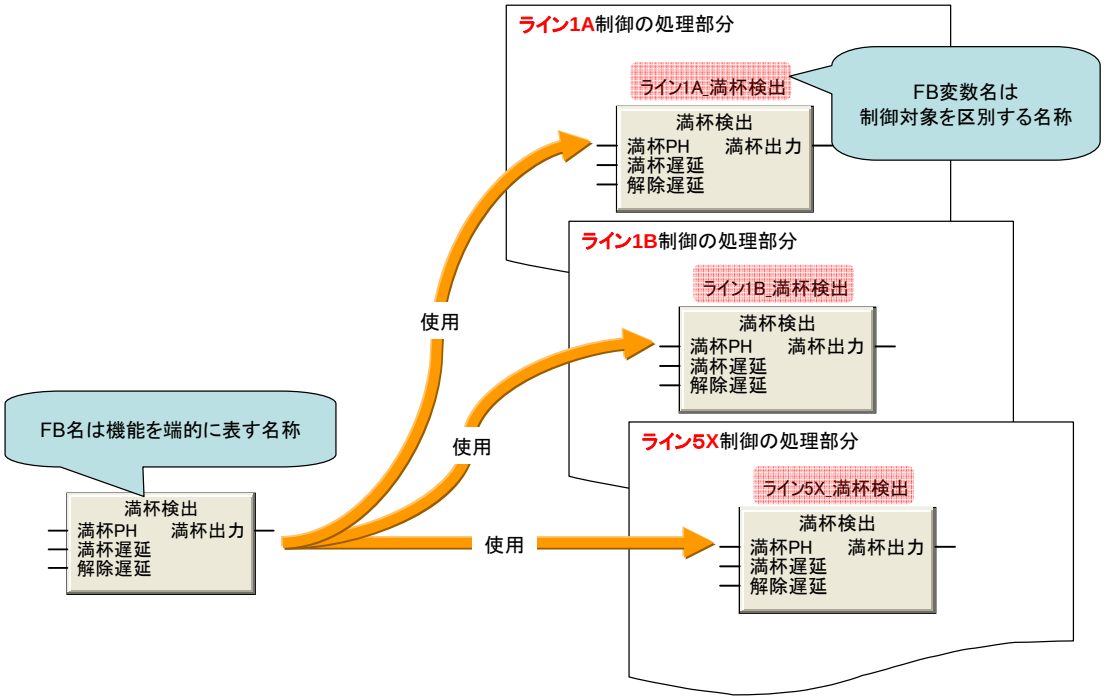


図25―FB使用時に命名するFB変数名の例

3.4.4　コメントの活用について

a)　**FBのヘッダコメント**　FBの作成者以外がその詳細(プログラムの内容)を知らなくてもそのFBを用いることができるように，必要となる情報をヘッダコメントとして記載する。FBのヘッダコメントとして記載が望ましい項目は，次による。

- 機能概要(用途，目的など)
- 入力変数の名称，データ形，意味，値の範囲
- 出力変数の名称，データ形，意味，値の範囲
- 外部変数の名称，データ形，意味，値の範囲，FB内部での値変更の有無
- 使用条件，制約条件
- 機能説明
- 変更履歴(作成・変更者，変更日付，変更内容)

これらの記載は，PLCベンダが提供しているメーカFBのリファレンスマニュアルを参考にするのもよい。

b)　**プログラムコード内のコメント**　プログラムのコード内に有益な情報をコメントとして記載することは，プログラム保守の観点から非常に大切なことである。

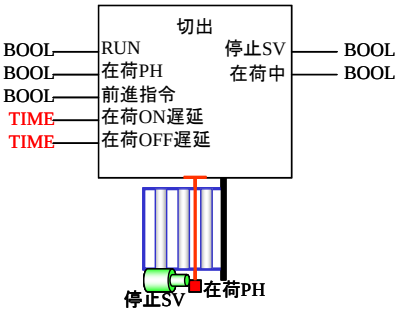
IEC 61131-3のプログラミング言語(LD言語，IL言語，ST言語，FBD言語，SFC言語)を活用し，かつ，端的に表現された変数名又はFB名によってプログラミングされたコードは既に可読性が高く，プログラムコードを読めば処理内容を理解しやすくなる。そのため，プログラムコード内のコメントは，“～する”という処理内容を記載するだけではなく，“～のため(の処理)”という実装の意図(目的及び理由，背景及び必要性など)をプログラムコードとともに記録しておくことがより有益といえる。

3.4.5　搬送システムにおけるプログラミングの例（切出FBの例）

この項では，FBの製作例として，搬送ユニットの切出機能を制御する“切出FB”のプログラム例を紹介する。

a)　切出FBの仕様

表10—切出FBの仕様

搬送ユニット名	搬送ユニット制御仕様	搬送ユニット制御FBの入出力仕様
切出	a）在荷PHによって荷を検出。 b）在荷ON遅延時間及び在荷OFF遅延時間の監視によって，ストッパ到達・到達解除を判定。 c）荷のストッパ到達によって，停止SVをON（荷の進行停止）。 d）荷の到達解除又は前進指令によって，停止SVをOFF（荷を進行）。 e）RUN信号の解除時は，停止SVを強制OFF。	

b)　切出FBのプログラム例

変数名及びコメント記述の工夫によって，作成されたプログラムが非常に理解しやすく実装されていることが分かる。

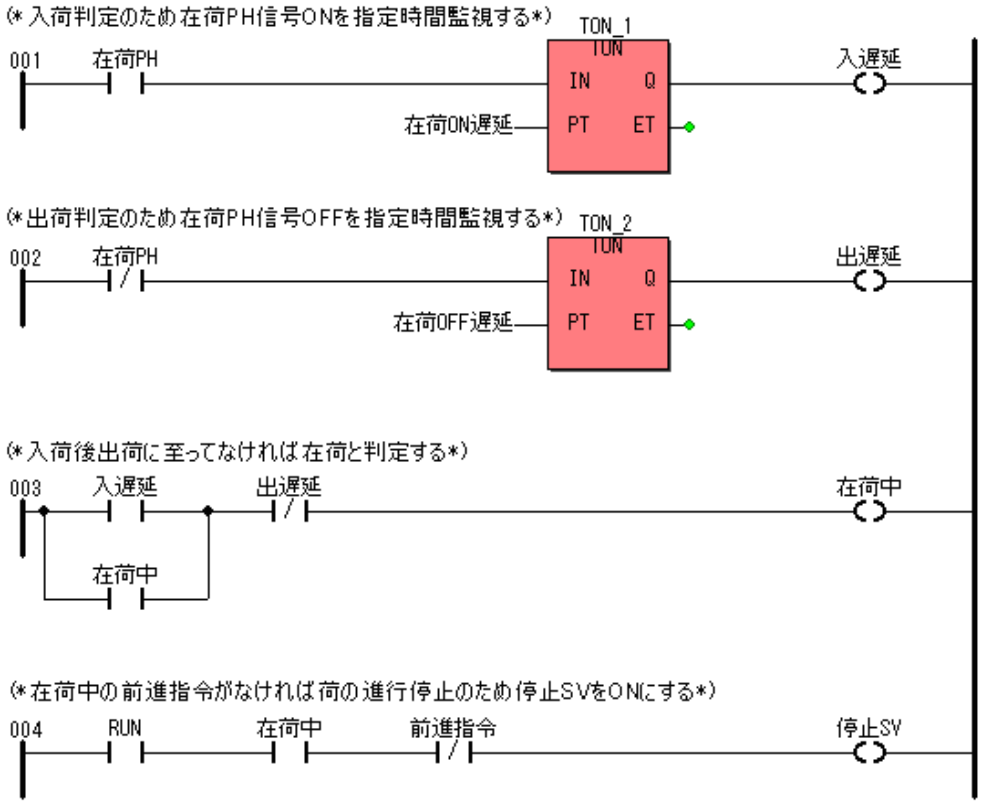


図26—コメント記述が分りやすい切出FBのプログラム例

3.5　テスト

テスト工程では，製作したアプリケーションS/Wの動作検証を行う。ここでアプリケーションS/Wは，FBが階層的に組み合わされた構造となっている。そのため，下層のFB（基本部FB）の単体テストから始め，上層のFB（応用部FB）へと順に組合せテストを行う。

まず，“分岐”FB以下の五つの基本部FBの単体テストを実施後，基本部FBを組み合わせた応用部FBをテストする。

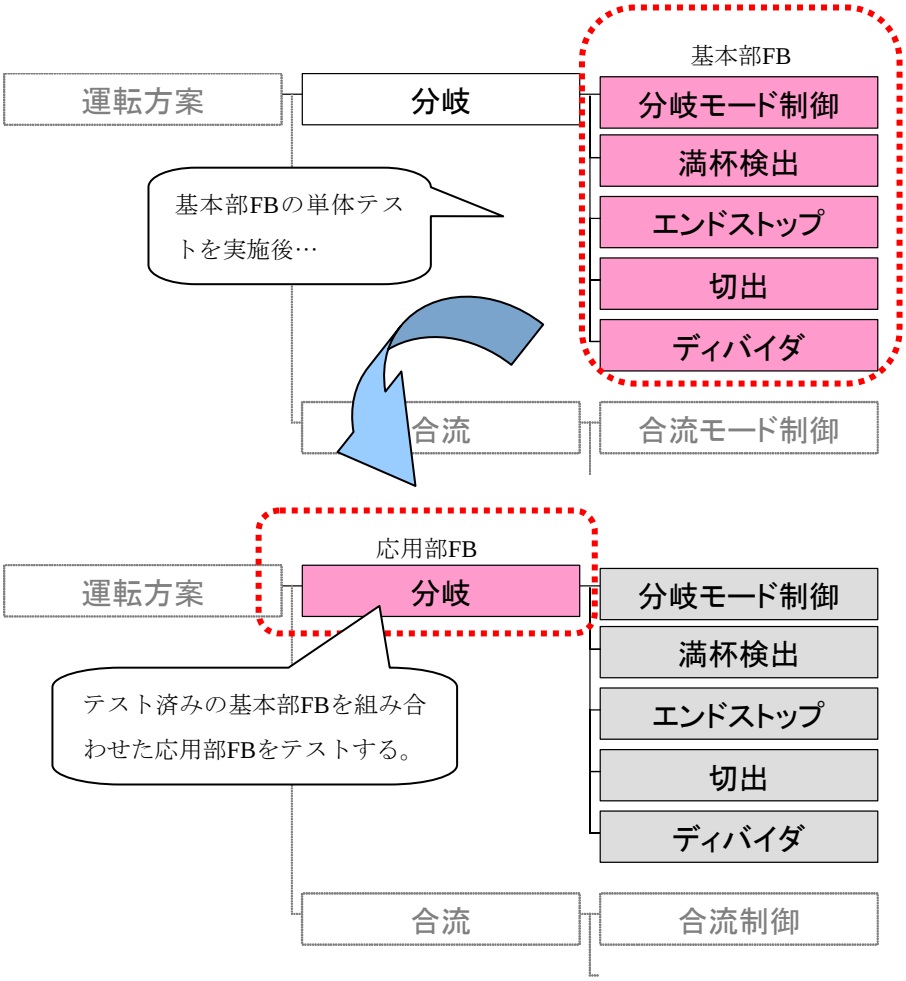


図27—分岐部分のテスト順序(基本部FBから応用部FBへ)

図29に示す“合流”FB以下のテストでは，一度テストを行ったFB(動作検証済みのFB)については，そのFBが再利用されていても，繰返しテストを行う必要はない。したがって，“合流”FB以下の八つの基本部FBのうち，上述の“分岐”FB以下の各FBのテストが既に実施済みである場合，“合流制御”FB及び“ベルトトランスファ”FBの二つの単体テストを実施後，基本部FBを組み合わせた応用部FBをテストすればよい。

こうして，動作検証済みのFBを再利用することは，テスト工数の大幅な削減となることがわかる。

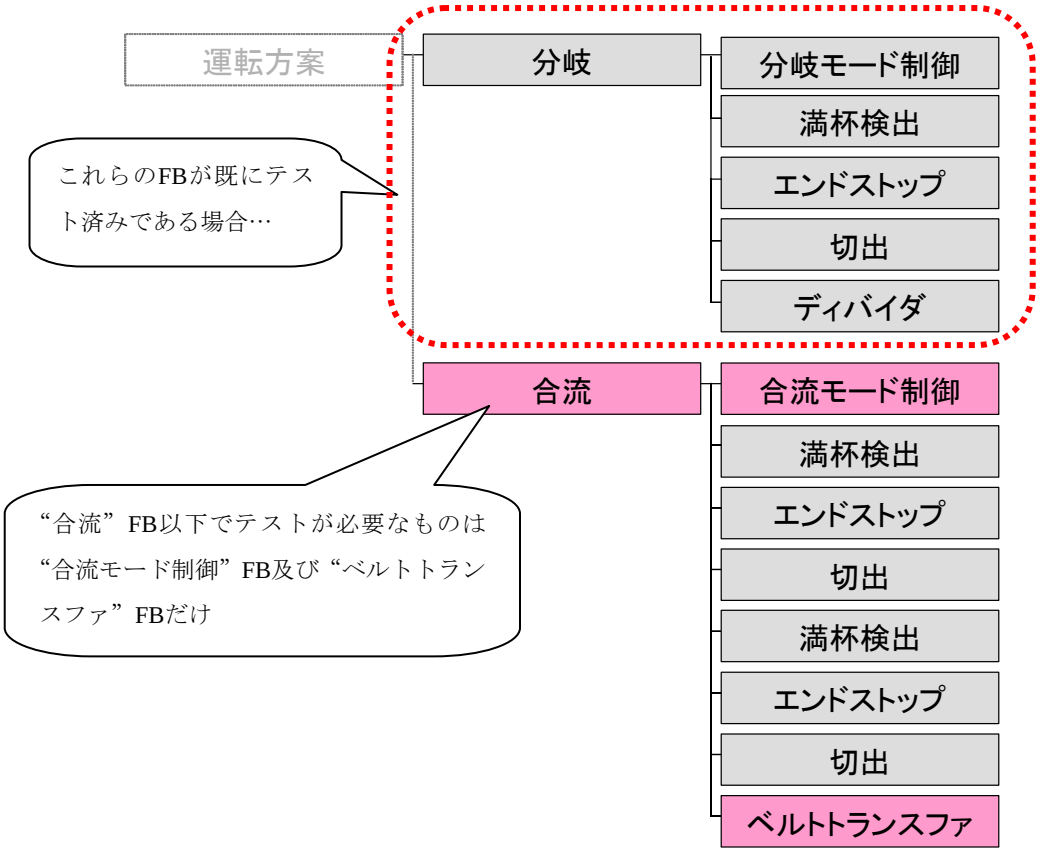


図28—合流部分のテスト順序(テスト済みのFBを含む場合)

3.5.1 FB単体テストの実施方法

a) テストケースの準備

FBの動作をテストする場合，あらかじめ何をテストするか(これをテストケースと呼ぶ。)を明確にせずにFBの動作をテストしようとする場合，無駄なテスト及び多くの確認漏れが生じて，確実な品質保証が困難になる。そのため，事前にテストケースを準備することが大切である。このテストケースの作成には，次の情報を元に作成するとよい。

1) FBの制御仕様

2) FBの入力値及び出力値の範囲

1)については，3.3の設計段階によって各FBの制御内容を明確化している。2)についても，3.4.4 a)のFBのヘッダコメントの記述によって各変数(入力変数，出力変数及び外部変数)の内容を明確化している。

このほか，必要に応じてFBの内部変数及び実行制御の様子を確認するテストケースも加えるとよい。

b) テストケースによるFB単体テストの実施

a)のテストケースに基づいて，テスト対象のFBを評価する。評価は，実際にPLC実機でプログラムを実行して動作検証を行うことができるが，PLCシミュレータを用いることでPLC実機を用いないで動作検証することもできる。PLC実機を用意する手間を省き，机上によって簡単に評価する場合に便利である。FB単体テストの実施方法について，次に二つの例を示す。

1) **暫定プログラムを用意しての単体テストの実施例**

ほかのプログラム要素の影響を受けることなくFBを単体テストするために，テスト対象のFBだけを抜き出した次のような簡単な暫定プログラムを作成する。

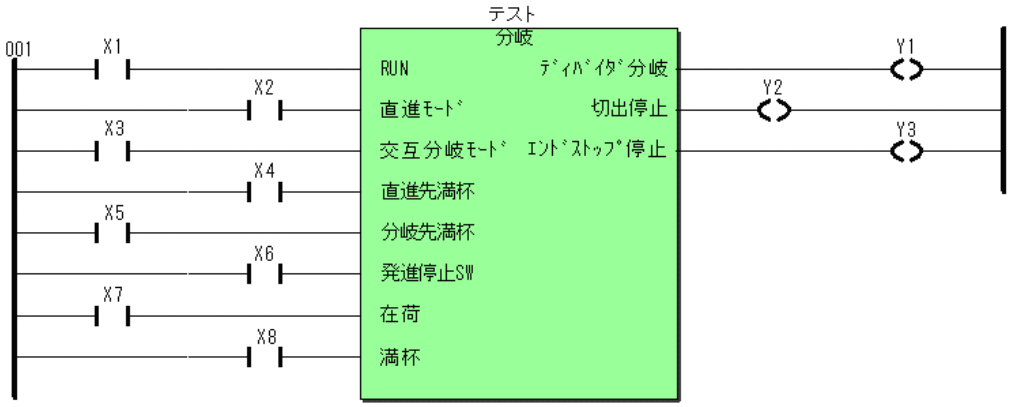


図29—暫定プログラムによるFB単体テスト

この暫定プログラムをPLCシミュレータで実行し，FBに模擬入力を与えて，期待される結果が出力されるかモニタリングすることで，a)で用意したテストケースの評価が行いやすくなる。

2) **アプリケーションS/Wに組み込んだままの単体テストの実施例**

ブレークポイント機能(一時停止機能)又はステップ実行機能を活用することで，テスト対象のFBを抜き出さず，アプリケーションS/Wに組み込んだまま単体テストを実施することができる。次の手順でテスト対象のFBを実際のプログラムに組み込んだまま動作検証を行う。

2.1) テスト対象のFBの手前で一時停止させる。

2.2) FBの入力変数に模擬入力を与える。

2.3) ステップ実行(又はFBの直後で一時停止)させてFBの出力及び内部の振る舞いを確認する。

3.6 ドキュメンテーション

ドキュメンテーションの主な目的は，システムの保守にある。

PLCシステムのS/Wに関する仕様書として，4.2の機能分析の結果をまとめた文書のほかに，プログラムそのものが活用できる。機能単位にモジュール化されたプログラムが可読性が高いプログラミング言語で記述し，更に十分なコメントが加わっているため，仕様書としての情報の量及び質を備えているからである。

3.7　保守

これまでに，この資料に述べたように，S/WがFBを用いた階層化設計がされている場合，様々な面で保守の効率が高くなる。その保守メリットの幾つかは，次による。

a)　**仕様変更又はトラブル原因調査の場合の影響範囲・関連部分を特定する場合**　システム改造に伴うS/W変更の場合，その影響範囲の特定が必要になる。また，システムのトラブル原因調査の場合も，該当する関連処理部分の特定が必要になる。

従来のラダープログラムは，一本(又は複数本)の長い巻物ように連なった複数の処理を羅列した構造となっているため，関連部分の特定の際にはその長い巻物を先頭から追いながら調べていくことがしばしばあった。また，ある処理を変更した場合にも，処理のら(羅)列からその変更に影響される部分を探し当てることは，とても手間がかかり，また，特定漏れを引き起こすことがあった。

一方，FBによる階層化設計がされたS/Wの場合は，次の特長をもっている。そのため，保守の効率が高くなる。

- 搬送ユニットのようなH/Wの制御は基本部FB，分岐又は合流といった機能をまとめ上げているは応用部FB，運転方案を司っているのは最上部のプログラムと，各部分の責務が明確に整理されているため，該当箇所の特定が行いやすい。
- また，FB及び変数は直感的に分かりやすく命名され，コメントの記載が手がかりになることから，プログラムの可読性が高く，関係部分が探しやすい。
- S/W構造が階層化されているため，あるFBの動作を点検する場合には，そのFBを構成するFBだけを掘り下げて点検すればよい(トップダウンアプローチによる点検)ため，効率がよい。

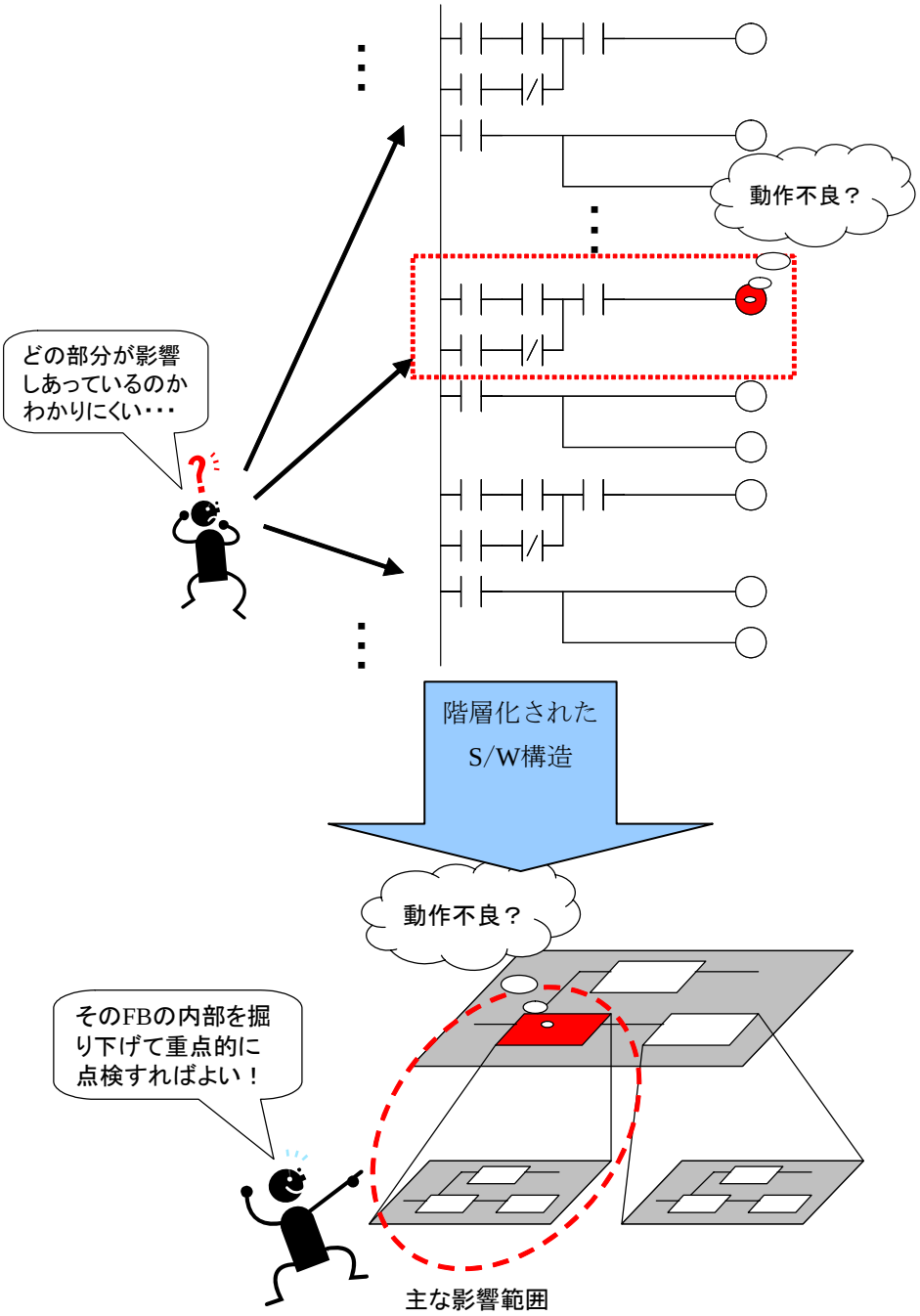


図30—階層化されたS/W構造の場合の動作不良箇所の絞り込み

b) **同じ機能を増設する場合**　既に作り込んだ機能(処理)をそのままの動作仕様で増設する場合，FBによる実装が大いに効果を発揮する。例えば，搬送システムにおいて，ラインBが既にあり，これと同じラインCを増築することになった場合，ラインBの合流・分岐の制御S/Wを再利用する。ラインC用に別のFB変数名を付けた“分岐”FB及び“合流”FBを運転方案プログラムに追加する。こうすることで，ラインCはラインBと動作仕様は同じだが，ラインBとは独立に動作する制御S/Wを簡単に増やすことができる。また，既に“分岐”FB及び“合流”FBが単体テスト済みであるため，ラインC増設時にこれらFBを改めて単体テストを行う必要はない。

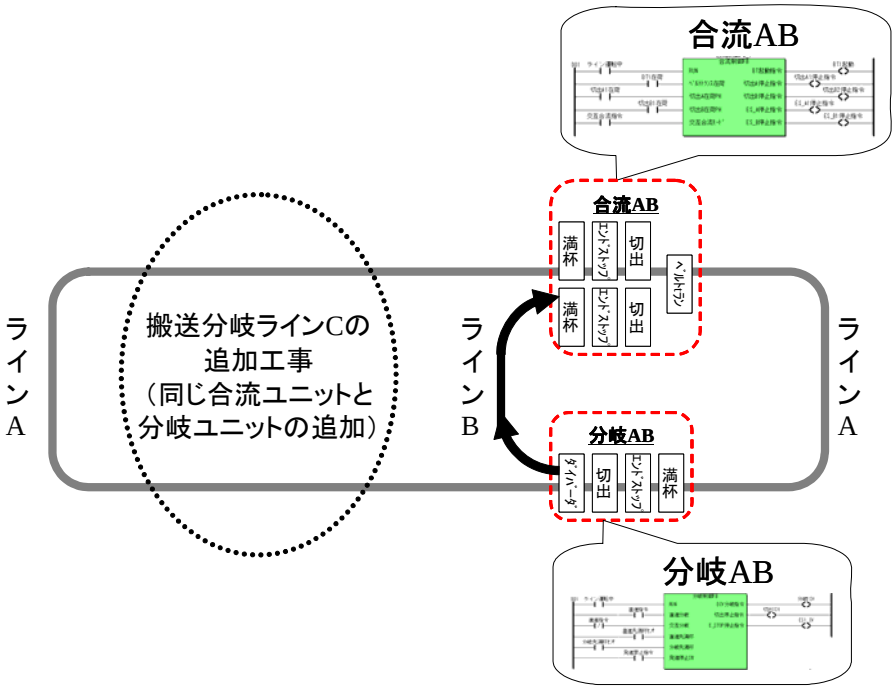


図31—搬送ライン増設前

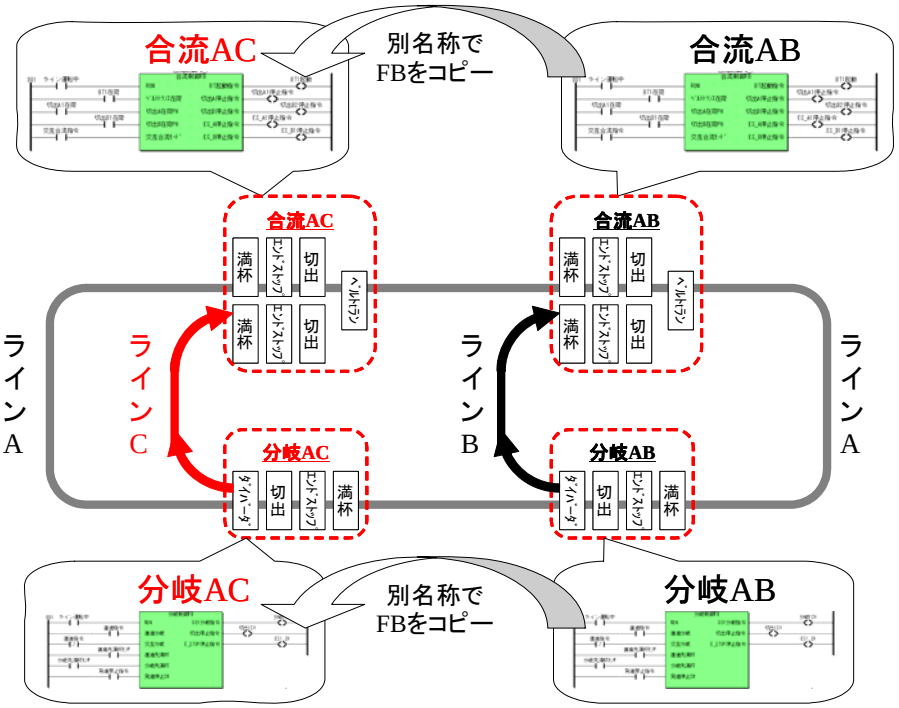


図32—搬送ライン増設後

c)　H/Wの仕様変更に対応する場合　システムの構成機器を別のものに置き換える場合，従来の機器と仕様が異なる場合がしばしばある。

例えば，すべての“切出”搬送ユニットを新しい仕様のものに一括変更する場合は，この“切出”搬送ユニットの制御を担当するFBのインタフェース（入力変数・出力変数・外部変数の種類及び役割）を変えることなくFBの中身だけを仕様変更に対応することができれば，ほかのプログラム部分を変更することなく簡単に機器の置換えに対応することができる。

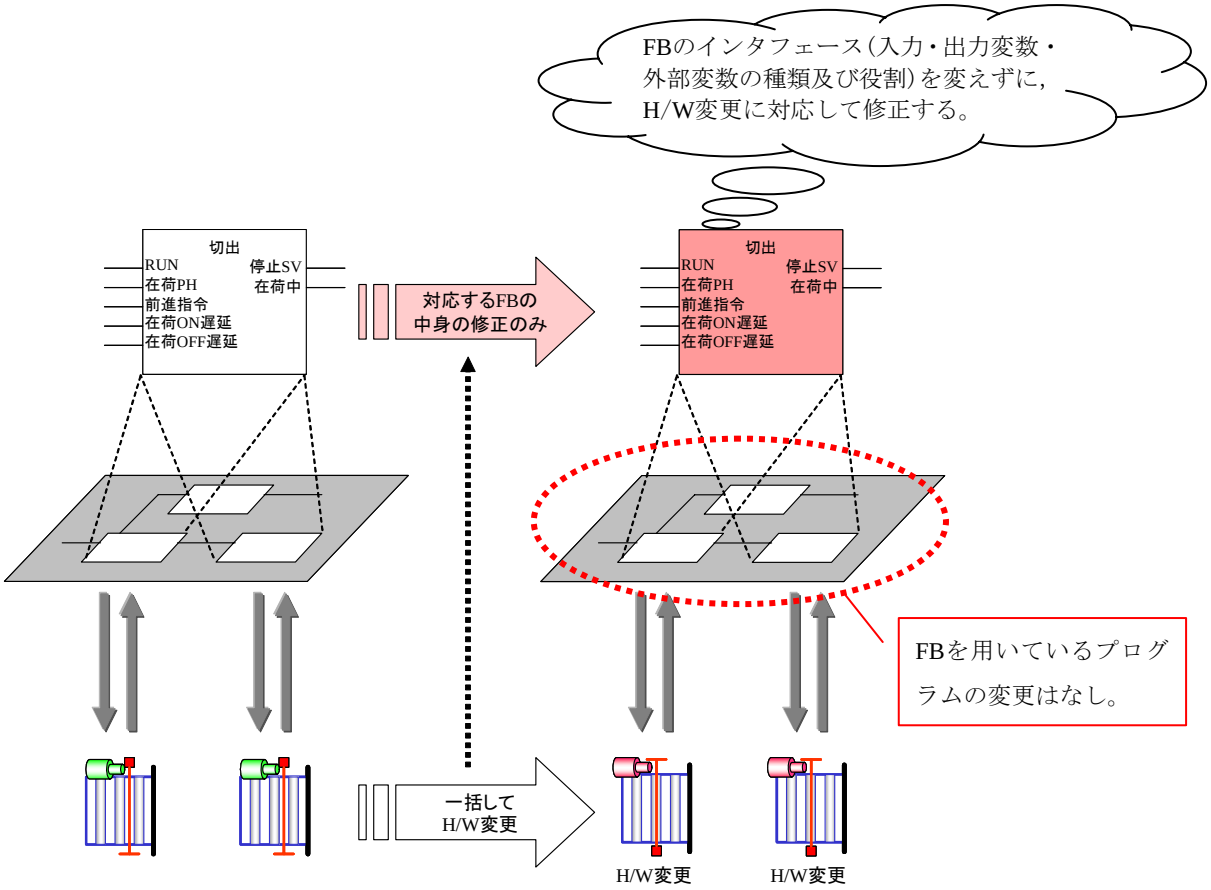


図33—H/Wを一括して変更する場合のS/W修正

また，一部の“切出”搬送ユニットだけを機器変更する場合にも，従来の“切出”FBと同じインタフェースをもち，この機器の変更に伴う新しい仕様の“切出_New”FBを新規追加できれば，該当する“切出”FBを“切出_New”FBに置き換えるだけで対応することができる(図34)。

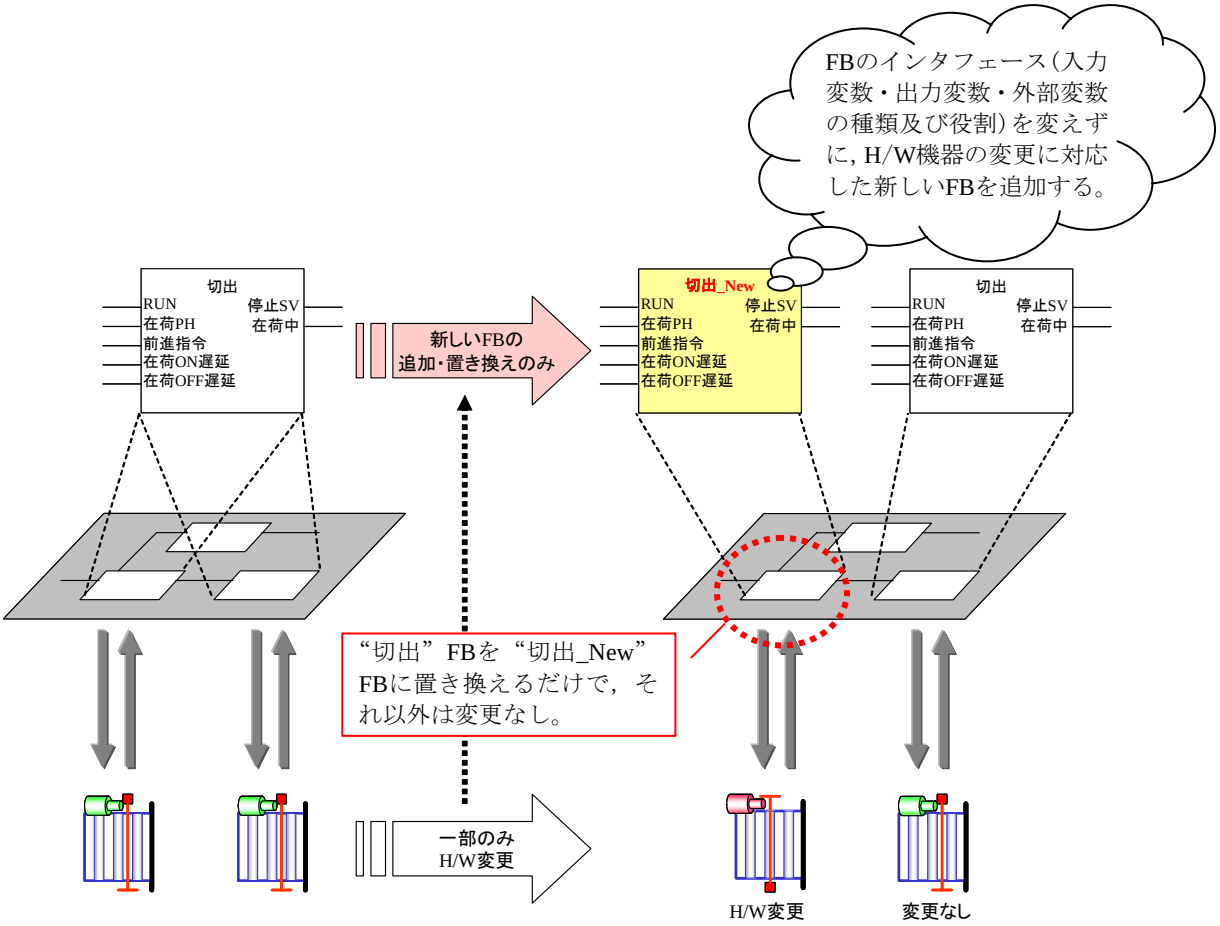


図34—H/Wを一部だけ変更する場合のS/W修正

4　まとめ

これまで説明してきた，PLCアプリケーションプログラムの開発効率の向上に向けた指針のポイントを，次にまとめて示す。

PLCユーザ（エンドユーザ，セットメーカ）の課題及びこれらに必要な対策は，**表11**による。

表11—生産設備ユーザの課題と対策

生産設備の課題	対策	S/W開発手法
生産能力向上	・段取換え作業効率の向上 ・MTTRの短縮化	・構造化・階層化設計 ・部品化及び再利用 ・最適な言語選択
開発コスト削減,	・設計手戻りの最小化 ・S/W資産の有効活用	
開発期間短縮	・S/W資産の有効活用 ・コンカレント開発	
保守コスト削減	・S/Wの属人化の排除	

これらのS/W開発手法を実施するためのポイントは，**表12**による。

表12—S/W開発手法のポイント

S/W開発手法	ポイント
構造化/階層化設計	・トップダウンによる構造分析フェーズの確保 ・設備の要求仕様の機能抽出，細分化及び共通化による機能構造の分析 ・設備への依存部及び非依存部に分けた構造化分析 ・運転法案及びユニット制御 ・装置制御，情報処理など ・構造化分析の結果に基づいた設計の実施
部品化及び再利用	・ファンクションブロック（FB）の適用及びボトムアップによるFBへの落し込み ・一つの制御をa）入力，b）制御内容，c）出力のセットによる仕様の明確化 ・機能又はH/Wの共通要素の抽出及び機能スイッチによる類似機能のFB化 ・設備に非依存なFBへの，外部I/O割付け禁止による汎用性の確保 ・所有FB，メーカ提供FBなどの管理及び利用 ・FB単位の品質保証によるS/W部品の標準化
最適な言語選択	・制御及び処理内容に適したプログラミング言語の選択 ・スキルに応じたプログラミング言語の選択 ・変数名及びコメントの付加よるプログラム自身のドキュメント化

これらのポイントを取り入れ，上記S/W開発手法を実践し易い仕組みのひとつに，IEC 61131-3（JIS B 3503）がある。この規格は，世界中のエンジニアが共通の言語でコミュニケーションでき，プログラムの部品化によって効率的にプログラムを生産できるよう規定されたPLCのための国際標準プログラミング言語である。

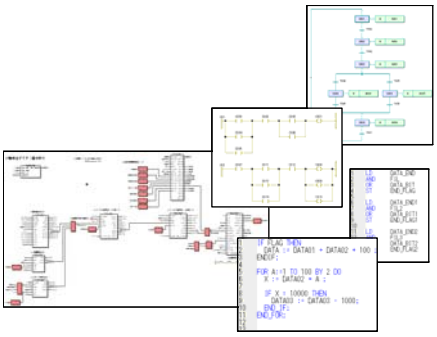
IEC 61131-3(JIS B 3503)には，次の特長がある。

a) **プログラムの構造化及び部品化**

- ファンクション及びファンクションブロック(FB)による部品化
- 変数によるプログラミングによる再利用性の向上
- プログラムとFBとの組合せによる構造化の仕組み

b) **用途に応じた言語の選択及び混在使用が可能**

- LD言語　単純なリレーシーケンス処理
- IL言語　プログラムメモリ制約の厳しい場合
- ST言語　数式演算処理，複雑な情報処理
- FBD言語　連続的なアナログ信号処理
- SFC言語　状態せん(遷)移に基づく順序制御



5　付録

5.1　IEC 61131-3（JIS B 3503）の解説

IEC 61131（JIS B 3503）は，PLC関連の規格であり，現在唯一のオープン化に対応した国際標準言語の規格として注目されている。オープンというは，この言語がPLCに依存しない世界共通のプログラミング言語だということである。これまで，制御装置というと各社ごとに専用の言語があり，採用する装置を変更するとその度に言語を習得しなければならなかったが，これからは，この規格の言語だけを習得すれば済む，ということになる。

国際規格IECで規格化されたPLCのプログラミング言語で日本の多くのPLCメーカが準拠をしている。日本ではまだ広く浸透していないが，欧米ではIEC 61131-3で記述できることがPLCの要求仕様となっている場合もある。

5.1.1　歴史

1993年：IEC 61131制定

1994年：OMAC　Paper発行　　（米3大自動車メーカの制御システム仕様）

1997年：JIS B 3503制定

世界的な状況として，1993年に制定されて以来，IEC 61131は欧州を中心に適用が進んだが，米国では適用が進んでいなかった。北米でも1994年のOMACペーパ（米3大自動車メーカの制御システム仕様）の発表を機に関心が急激に高まっている。これは，OMACペーパでIEC 61131-3対応のプログラミング言語の使用が規定されたためである。これまで，専用言語しかサポートしてこなかったPLCメーカがツールをIEC 61131-3対応に変更してきている。国内では，1997年にJIS B 3503として制定されている。

5.1.2　特長

a)　5言語（4言語・1要素）

b)　ユーザ定義ファンクション・ファンクションブロック

c)　ユーザ定義データ形（配列・構造体）

d)　オブジェクト指向対応

e)　厳しい文法・データ形チェック

5.1.3　用語

IEC 61131-3（JIS B 3503）の用語について簡単に紹介する。今まで，一般的に用いてきた用語から大分変更になっており，違和感がある可能性を考え，従来の呼び方と比較して説明する(表13)。

表13—IEC 61131-3(JIS B 3503)の用語と説明

IEC 61131（JIS B 3503）用語	従来の呼び方	補足
コンフィグレーション	システム	個々のコントローラは，リソースと表現している。コントローラの集合をコンフィギュレーションと呼ぶ。
リソース	コントローラ（CPU）	
タスク	（タスクの）優先度	タスクは，デフォルト，サイクリック及びイベントの3タイプがある。デフォルトは，普通の連続スキャンに当たる。これらは，従来での優先度に対応する。
プログラム	タスク	プログラムはそれぞれ独立しており，タスクに割り付けていく。これらのプログラムが従来のタスクに相当する。
ファンクションブロック（FB ） ファンクション	サブルーチン	ファンクション及びファンクションブロックはプログラミング言語で書いた機能ブロックであり従来のサブルーチンに相当する。さらにファンクションブロックは内部データを持つことができる。
グローバル変数	レジスタ	グローバル変数は，プログラム間で共有される変数である。これは，従来のレジスタに当たる。これに対して，プログラムの内部だけで用いられる変数をローカル変数と呼ぶ。
アクセスパス	通信，コモンメモリ	アクセスパスは，リソース間又はコンフィグレーション間でデータを交換する道筋を表現している。これには，従来の通信，コモンメモリが相当する。

5.2　プログラム

プログラムは，お互いにデータを交換できる数多くのファンクション及びファンクションブロックを接続し構成している。プログラムは，I/O変数に対して読出しも書込みもでき，ほかのプログラムと通信することもできる。プログラムは実行する単位である。

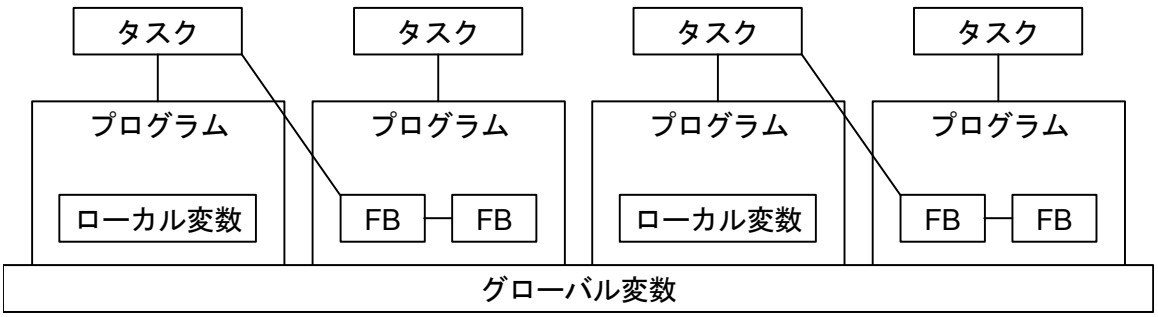


図35—プログラムの構成要素とその関係

5.3　ファンクションブロック(FB:Function Block)

ファンクションブロックは，制御プログラムの一部をパッケージ化したものである。したがって，同じプログラムの別々の部分においても，又は異なるプログラム及びプロジェクトでも再利用することができる。ユーザ定義ファンクションブロックはプログラム部品と考えて頂けばいいのだが，これはオブジェク

ト指向対応になっており，一旦作成したものを内部データも含んで繰返し用いることができる。そして，ファンクションブロックの中の環境はそれぞれ独立しているので，ほかのファンクションブロックの実行の影響を受けることはない。これは，同一の変数名を使用していても構わない，ということである。また，変数名を用いてプログラミングができるので，使用者は，メモリの割付けを一切考慮する必要がない。これは，レジスタでのプログラミングに慣れた方には少し違和感があるかもしれないが，レジスタの配置及び残り容量を気にすることがないので，とても楽なプログラミングスタイルである。

ある機能単位の処理をまとめ，後に流用を考えた部品としてFB化にすることによって可読性，保守性の良さ，ひいてはソフトウェア生産性の向上などのメリットが生まれる。

FBとファンクションとの違い(出力の多値性及び内部保持変数)は次のとおりである。

a) ファンクションには，内部保持機能がないため，実行する度に，同じ入力に対して，常に同じ結果を生成する。それに対し，ファンクションブロックには，内部保持機能があるため，同じ入力に対し，常に同じ結果を生成するとは限らない。

b) ファンクションは一つの出力しかもてないが，ファンクションブロックは複数の出力をもつことができる。

表14—ファンクションとファンクションブロックの違い

	内部保持機能	出力	同じ入力に対して	例えば
ファンクション	なし	1つ	常に同じ結果	加算等の算術演算
ファンクションブロック	あり	複数	同じとは限らない	カウンタ，ディレイタイマ

例えば，出力値を，低い値からより高い目標値まで，与えられた比率で着実に増やしていくランプ・ファンクションブロックについて考えてみる。ランプ比率及び最終的な目標値を定義する入力値は，変更されずに固定されているにも関らず，出力値は，ランプのファンクションブロックが実行される度に期待する目標値に達するまで，少しずつ増加していく。こうしてファンクションブロックは，出力変数及び内部保持機能の両方に値を保存することができるのである。

5.4 言語の解説

5.4.1 LD，IL，ST，FBD及びSFCの概要

プログラミング言語として，図式言語3種(LD/FBD/SFC)及びテキスト言語2種(IL/ST)の計5種類が決められている。

・ LD(ラダー　ダイアグラム：Ladder Diagram)
・ IL(インストラクション　リスト：Instruction List)
・ ST(ストラクチャード　テキスト：Structured Text)
・ FBD(ファンクション　ブロック　ダイアグラム：Function Block Diagram)
・ SFC(シーケンシャル　ファンクション　チャート：Sequential Function Chart)

SFCは，状態のせん移を制御するだけで，演算機能並びに入出力機能がないために，言語ではなく要素という取扱いなので，正確には4言語1要素ということになる。

IECのプログラミング言語は，文法規定がかなり厳しくなっている。例えば，これまで命令の入力は，レジスタということで，そのレジスタが倍長で用いられているものの上半分でも下半分でも構わずに単長のワードとして用いることができるケースがあったが，IEC 61131-3では変数はデータ型を含めて厳しく検査しているので，そのようなことはできない。これは，文法上の検査を厳しくして，不用意なデータ使用

によるプログラミングミスを防ごうとしているからである。これは，初めのうち，面倒に感じるかもしれないが，慣れれば，デバッグが困難なミスを初めから排除してくれるので，多大な効果が期待できる。

5.4.2　適用用途の向き不向き

a) LD　リレーシーケンスの置換え　PLCに慣れ親しんだ方に。

　LDは，リレーを用いたロジックの設計技術に基づいており，左側に垂直の母線をもち，回線上への接点へ，仮想的に電源を供給している。電気回路図の実配線図を仮想的に表現している。

b) IL　アプリケーションの小形化　アセンブラに慣れ親しんだ方に。

　ILの基本構造は，非常に単純で，学習するのも簡単である。判定のポイントが少なく，プログラム実行時の流れの変化が少ないような，規模が小さく簡素な課題の解決には理想的である。

c) ST　構造化プログラミングを支援するハイレベルなテキスト言語で，PASCALに非常に似た言語構文をもっている。パソコン高級言語に慣れ親しんだ計算機技術者に。

　STは，変数への値の割当てを始め，ファンクション及びファンクションブロックの呼出し，式の作成，選択文の条件評価，反復(繰返し)など，総合的な構文を用意しているので，かなり自由なスタイルで文を書くことができ，タブ，改行キャラクタ，コメントなどを，キーワードと識別子との間ならどこでも必要なあらゆる場所に挿入することができる。

d) FBD　ファンクションブロックによって，信号及びデータの流れを描くグラフィック言語である。DCSに慣れ親しんだ方に。

　ファンクション及びファンクションブロック並びにプログラムの動作を，互いに接続した一連のブロック図として表すことができるので，処理要素の間の信号フローがどうなっているかを視覚的に見ることができる。電子回路図に描かれている信号フローに似ている。

e) SFC　アプリケーション記述言語　制御システム全体の動作を表現するのに便利。

表15—IEC61131 5言語の特長と選択指針

言語	特長	選択指針
LD	電気回路図の実配線図を仮想的に表現	PLCに慣れ親しんだ方に。 シーケンス制御に。 インタロック回路に向く。
IL	基本構造は，非常に単純	アプリケーションの小型化に向く，
ST	構造化プログラミングを支援するハイレベルなテキスト言語	パソコン高級言語に慣れ親しんだ計算機技術者に。 論理演算，数式演算及びデータ処理に向く。 条件分岐，繰り返しの記述に向く。
FBD	信号及びデータの流れを描くグラフィック言語	DCSに慣れ親しんだ方に。 計装，ループ及びループバック制御。 FN/FBの構造化/階層化記述に向く。 データフローを明確したい。
SFC	アプリケーション記述言語	制御システム全体の動作を表現するのに便利。 状態せん移の記述に向く。

5.4.3　5言語比較

同じ処理でどう変わるかをSFC以外の4言語を比較する。

変数Aと変数Bとの反転値の論理和を変数Cに代入するというものである。

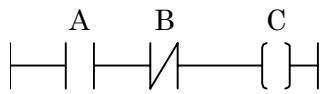
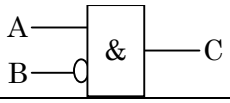
<table>
<tr>
<td>・ LD
 
</td>
<td>・ IL
LD　A
ANDN B
ST　C</td>
</tr>
<tr>
<td>・ ST
C := A AND NOT B</td>
<td>・ FBD
 
</td>
</tr>
</table>

図36—4言語の同一処理記述比較

・ SFC

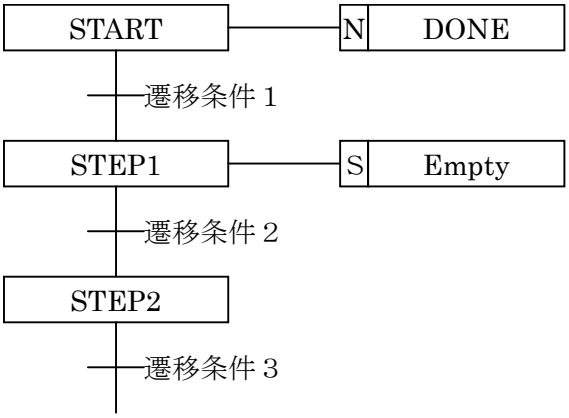


図37—SFC言語の記述例

5.5　変数

IEC 61131-3（JIS B 3503）ではデータを変数として扱う。仕様用途に応じて，表16のように分類する。

表16—変数の種類と内容

変数の種類	内容
ローカル変数 （内部変数）	プログラム及びファンクション・ファンクションブロック内で有効な変数。
入力変数	プログラム及びファンクション・ファンクションブロックに対し外部から与えられる変数。
出力変数	プログラム及びファンクション・ファンクションブロックから外部に出力する変数。
入出力変数	プログラム及びファンクション・ファンクションブロックで外部と入出力する変数。入力と出力とを兼ね合わせた変数。
グローバル変数 （外部変数）	プログラム及びファンクション・ファンクションブロック内外で有効な変数，それぞれの間で共有するときに用いる。

5.5.1　変数の形

IEC 61131-3（JIS B 3503）では，用途に応じたデータ形を，**表17**のように規定している。

表17―変数の形

分類	データ形	内容	ビット	範囲
整数形	INT	整数	16	－32768～－32767
	DINT	倍精度整数	32	-2^{31}～$+2^{31}-1$
	UINT	無符号整数	16	0～65535
実数形	REAL	実数	32	$\pm10^{\pm38}$
	LREAL	倍精度実数	64	$\pm10^{\pm308}$
日付形	DATE	カレンダ日付		カレンダ日付を保持
時刻形	TIME_OF_DAY	時刻		時刻を保持
持続時間形	TIME	持続時刻		イベント発生後の持続時間を保存
文字列形	STRING	文字列		テキスト情報を保持
ビット形	BOOL	1ビット列	1	0/1
	WORD	16ビット列	16	0000～FFFF
	DWORD	32ビット列	32	00000000～FFFFFFFF

本資料の最新版の入手は・・・

本資料の最新版は，電子データダウンロードにて入手が可能です。JEMAのウェブサイトのオンラインストアにおいて無償公開出版物としてダウンロードが可能です。
JEMAウェブサイトURL：http://www.jema-net.or.jp/

本資料の内容に関するお問合せは・・・

社団法人 日本電機工業会　技術部　技術課
TEL：03-3556-5884/FAX：03-3556-5892

平成22年2月1日 発行　　〒102-0082　東京都千代田区一番町17番地4
発 行 所　　社団法人 日本電機工業会

技09-01