SFCの現状と課題

－新しいシーケンス制御の記述法－

本　田　雅　夫

目　次

1．はじめに

SFCはSequential Function Chartの略語であり、日本語訳すれば「シーケンス状態遷移図」と言い表すことができるPC（プログラマブル・コントローラ）の新しい記述方式である。

従来、シーケンス制御のハードウェアの主流はリレー回路でしたが、接点の接触信頼性、配線作業量、制御内容の変更、定期的保守等幾つかの問題を抱えていた。リレー回路はこのような問題や、多くの欠点があるにもかかわらず、他に有効なハードウェアがなかったため、シーケン

ス制御の主流になっていた。1968年米国のGM（ゼネラルモータ）社の開発要求によりPCが出現し、大型・複雑化したシステムに対する信頼性の解決と複雑多様化した制御システムの制御装置の改造作業の容易化が実現した。

その結果、現在PCは、FA（Factory Automation）の中核機器として幅広く利用され、省力化・自動化に不可欠な電子応用機器と言われている。その利用形態は、工場全体を制御するシステマティックなもの（各種のPC通信）から各種分散機械を個別に制御するスタンドアーロンタイプ（単独仕様）のものまでさまざまな形態にわたっている。

PCを使用したPC制御システムの記述表現法（PC言語）には幾つかあるが、中でもグラフィック形のラダーダイヤグラム（PC言語の一つ＝以下ラダー）といわれる記述表現法が主流になっている。SFCはペトリネット（現象の遷移を状態図で表現する記述方式）の概念をベースにPC言語に応用したGRAFCET（欧州を中心に使用されいてるグラフィック言語）を発展させ、実用化したものである。また、SFCは、シーケンシャルな事象の制御を表現するフロー手法であり、PC言語と組み合わせることによりプログラムを実行する制御単位の「要素」といえるものである。SFCは、ラダーのソフト開発、運用、保守にわたってデメリットとなっていた①分かり難い②保全性が悪い③可視性が悪い④ソフト工学的なプログラムでない等をメリットに変え、高速応答やプログラムの標準化・構造化等を実現させたプログラミング手法と言える。（付録１，２）ヨーロッパでは、サードパーティ製のSFCツールが市販されるなど、SFCの実用化が進んでいる。さらに規格面でもSFCを規格した国際規格である IEC 1131-3（Programmable Contoroller Programming Language)、IEC 848（Preparation of function charts ofcontrol systems）及びJIS－B 3503の規格化が進んでいる。（付録３）

最近、シーケンスの制御システム構築においてアプリケーションソフ

ト容量の増大や設計効率・品質の向上・メンテナンス等の大きな変化が現れている、アプリケーションソフト容量については、PC を用いた設備・機械の設計開発に携わる業種では、自社製品の市場における他社との競争力強化のため、設備・機械の高機能化、付加価値の創出に迫られている。これに伴い PC のアプリケーションソフトのプログラム容量も確実に増え続けている。その理由として、設備・機械自体の基本機能の高機能化・多機能化だけでなく、設備、機械の信頼性の向上やユーザインタフェースの重要性が高まり、ソフトウェア的なトラブルシュート、ロギング等のソフトロジックが増加していることが考えられる。また近年、現場の情報化の飛躍的な発達に伴うネットワーク化や、PC の周辺制御の高度化による周辺コントローラとの協調化に伴い、情報ハンドリングのための処理が増加し、かつ制御ロジック自体も複雑化してきている。

また、設計効率・品質の向上，メンテナンスについては、昨今の構造不況、円高、人件費の高騰等の影響により、さらなるコストダウン、納期の短縮化、メンテナンスの効率化要請が強まっており、多くの PC ユーザは、設備・機械に占めるハードウェアコストより比率的に高まりつつある人件費を抑えるために、保全工数の削減を設計段階で行うという判断がなされている傾向がある。そのための方策として、ソフト設計技法の取り込みやソフトウェアのモジュール化による再利用性の向上等を積極的に試行することにより、ソフト設計効率や設計品質の向上を図ろうとする動きが全体的に目立ってきている。

一部の PC ユーザーではこの問題は深刻化しており、制御ソフトの領域においてはまだ発展途上と考えられるオブジェクト指向技術を PC のアプリケーションプログラムに取り込もうとする動きすら出てきている。

この様なことから、そのソフトの生産性及び信頼性又は品質向上（メンテナンス性を含む）を図るためにはどうあるべきかと言う点に視点を置いて、PC 言語の中での SFC の持つ意味・位置づけ、SFC の制御系の

記述の有効性、他の記述法に比べてSFCを使うメリットがどこにあるかについてSFCの普及の観点から問題点と今後の動向について述べる。

2．SFC とは

2-1．PC 言語と規格

国際電気標準会議（IEC ＝ International Electrotechnical Commission）ではPCの国際的な規格を審議しており、その中にPCのプログラミング言語が含まれている。

IEC 1131規格は、次の5つの部分から成り立っている。

(1) Part1: General Information

(2) Part2: Equipment and Test Requirements

(3) Part3: Programmable Contoroller Programming languages

(4) Part4: User Guidelines

(5) Part5: Communication

IEC規格では、言語の基本的な命令機能や入出力信号の分類、記号等のソースレベルでの標準化を行っており、PCの中の命令については命令名称、記載方法等の厳格な制限は加えていない。規格は最低限の遵守項目を規定しているものの、現在のPCにはない高度な処理機能なども含まれ、その拡張のためのルールも規定してあり、将来に対し、柔軟に対応できるようになっている。この中のPart3（IEC 1131-3）が、現在Industrial Control Programmingの世界で、標準開発言語として定着しつつある言語体系である。IEC 1131-3規格は、4つの言語（IL，ST，LD，FBD）体系と1つのSFC要素から成り立っており、その他に、Function, Function Blocks, Function Blocks Instanceと言うアプリケーションの各パーツを利用したり、実行し構成するメカニズムも定義されている。アプリケーションの各パーツは、それに適切な言語でプログラムでき、アプリケーション全体が単一の実行プログラムに間断なく

リンクできるようになっている。表1はIEC 1131-3の概略である。

なお、IEC 1131-3を翻訳したものが、JIS－B 3503（IEC規格の改定が予定されていたため正式な制定を保留していた。）である。SFCはPCの言語ではなく、PC言語で書かれるプログラムの可読性を高めるための共通要素の一つ（SFC Element）として規定されている。

表1　IEC 1131-3の概略

テキスト形	文字・記号を使い、式やリストにより制御内容を"文章"形で記述する言語。
	① ブール代数：出力機器が動作するための（AND, OR, NOT)条件をブール代数による論理式で制御内容を記述。 ② ＩＬ(Instruction List＝命令リスト):アッセンブラの様なローレベル言語で、命令とデバイスを一つ単位となるプログラム行とし、この行を演算実行順に配列した一連のリストから構成して制御内容を記述。通常、ＩＬはＬＤと１対１の関係にある。 ③ ＳＴ(Structured Text＝構造化テキスト)：ハイレベルのブロック構造化言語で、制御の論理演算や数値・データ処理の内容を文書形式で表現し、プログラムの実行順序に関係なく処理内容を記述。通常、ＦＢＤ言語をテキスト形言語に変換した場合ＳＴ言語になる。
グラフィック形	回路図で制御内容を記述し、視覚的に分かり易く、理解し易い
	④ ＬＤ(Ladder Diagram＝ラダー図)：信号の論理組み合わせの演算内容をリレーのシンボルにより表現し、ａ接点・ｂ接点のシンボルとこれらの直・並列接続で記述。 ⑤ ＦＢＤ(Function Block Diagram＝ﾌｧﾝｸｼｮﾝ･ﾌﾞﾛｯｸ図)：AND・OR 等の論理演算ブロックの組み合わせで表現。論理のみの機能を持つものは論理回路図と呼ぶ。 ⑥ ﾌﾛｰﾁｬｰﾄ（Flow Chart)：処理の手順を流れ図で記述。 ⑦ ＳＦＣ(Sequential Function Chart)：信号組合せの論理演算、数値演算、データ処理にLD, IL, FBD, ST言語を付加し、これらの処理の順序をグラフィック記述したプログラムの実行制御単位の要素。
テーブル形	制御対象の動作順に入力信号の組合せパターンを表（テーブル）形式で記述。
	⑧ デシジョンテーブル（Decision Table）

このほかに、“GRAFCET”がそのまま国際規格となった IEC 848（FC＝Preparation of function charts of control systems）がある。FC は、ステップ及びトランジションの記述が、制御対象の名称を用いて記述できる様になっていて、要求仕様書（運転方案）作成ツール的になっている。SFC は IEC 848をベースに作成されたが、大きく変わった点は、ステップ、トランジションの中味を記述するのに PC 言語を用いて記述するように定められたことにある。

2-2．原理と各要素

SFCの基本的な表現ルールはIEC1131-3の規格により、プログラムを構成する基本要素（命令）は次の 6 つがある。

①“処理工程”を表すステップ
②“初期処理”を表すイニシャルステップ
③“移行条件”を表すトランジション
④“具体的な処理内容”を表すアクション
⑤“別の所に飛ばしたり入れたりすること”を表すジャンプ／ジャンプエントリー
⑥“遷移の道程”を表すリンク

基本要素を使って SFC を記述すると、図 1 のようになる。

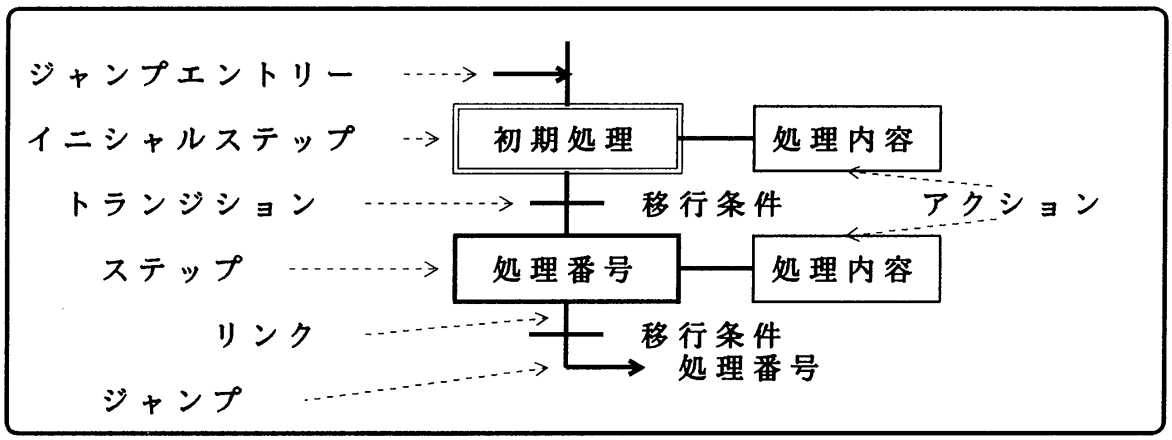


図 1　SFC 構成図

図2は、SFC全体構成概略図である。

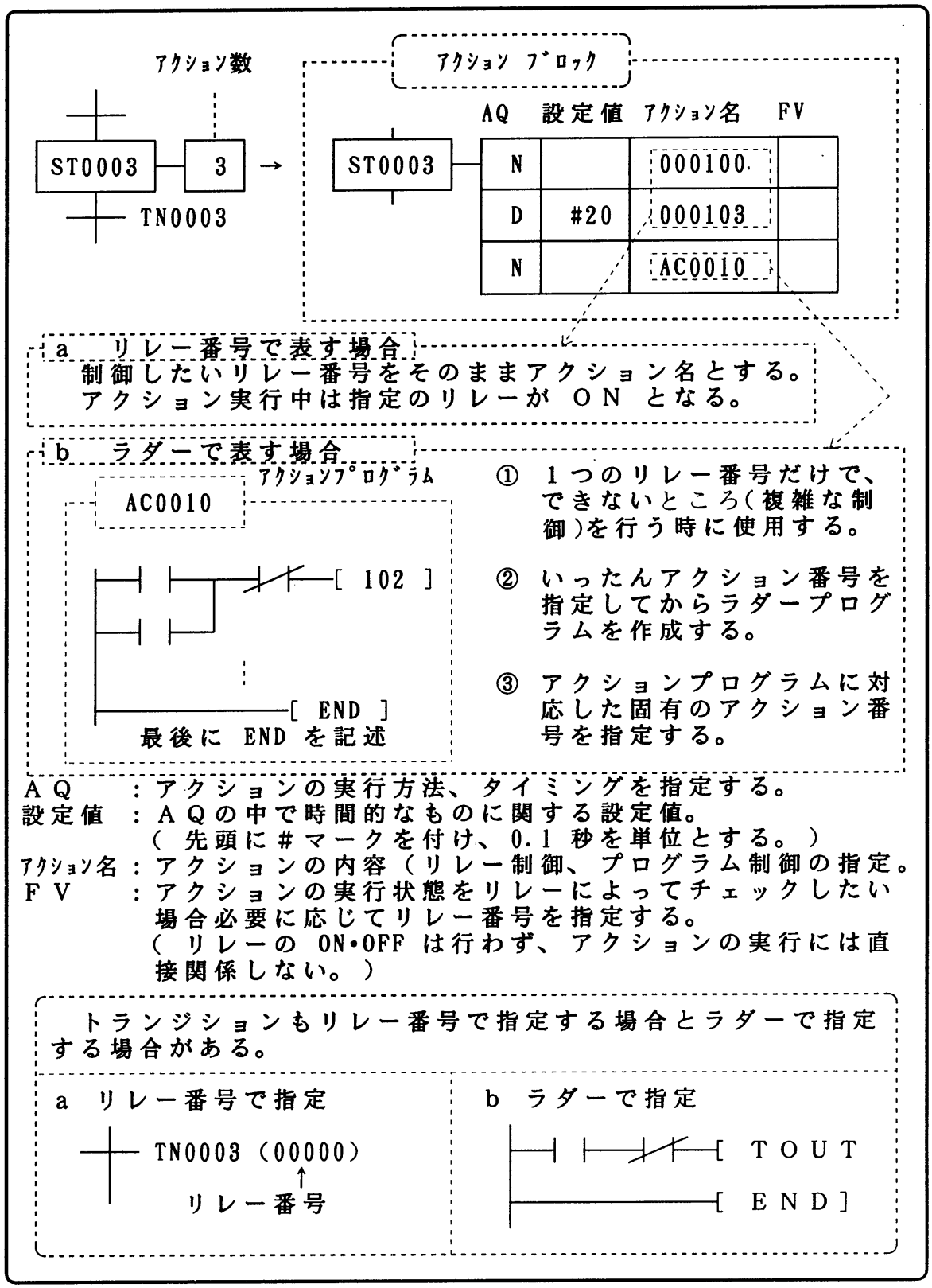


図2　全体構成の概略

2-3．基本的動作パターン

プログラムを構造化し、工程の流れをそのままプログラムに反映するために“SFC”の基本動作パターンには、次の５種類がある。

(1)　直列実行（図３参照）

(2)　並列分岐（図４参照）

(3)　並列合流（図５参照）

(4)　選択分岐（図６参照）

(5)　選択合流（図７参照）

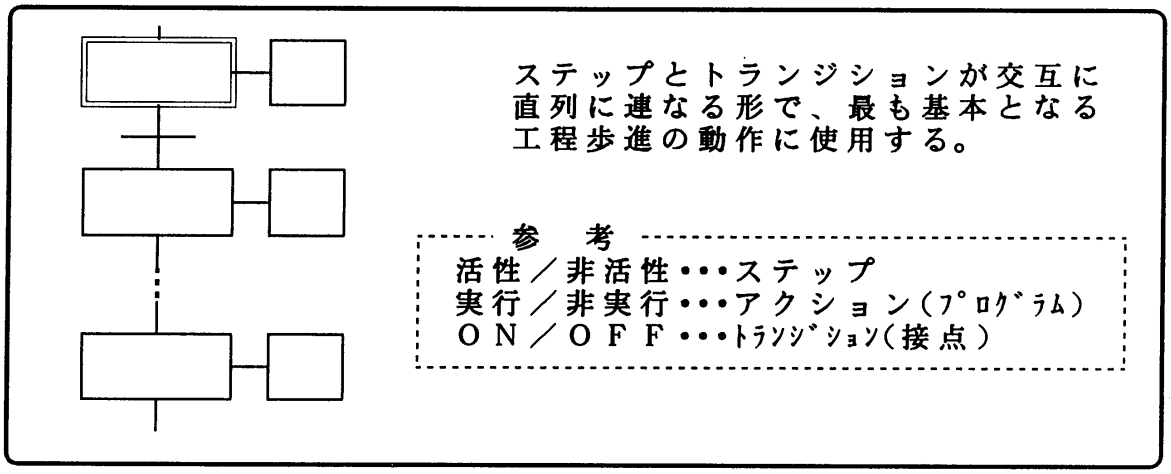


図３　直列実行

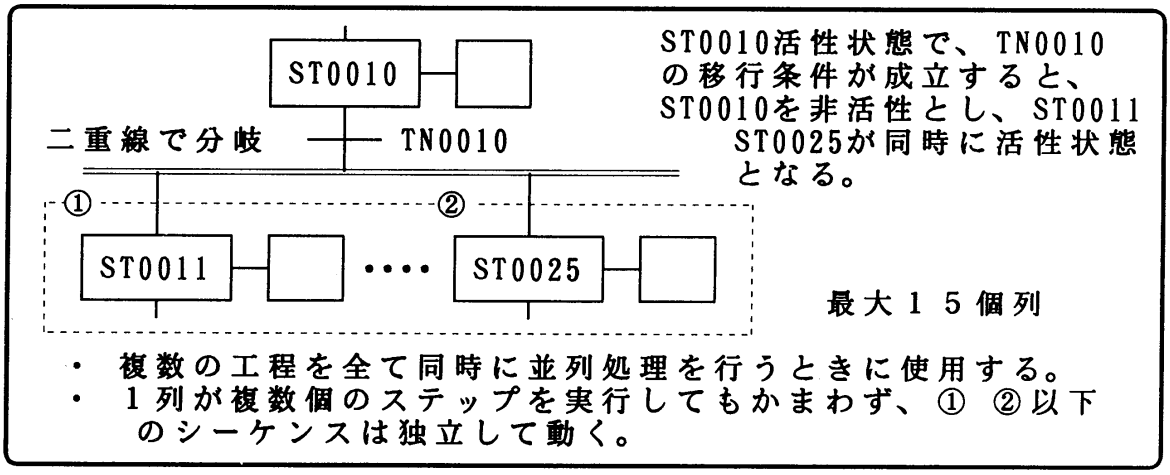


図４　並列分岐

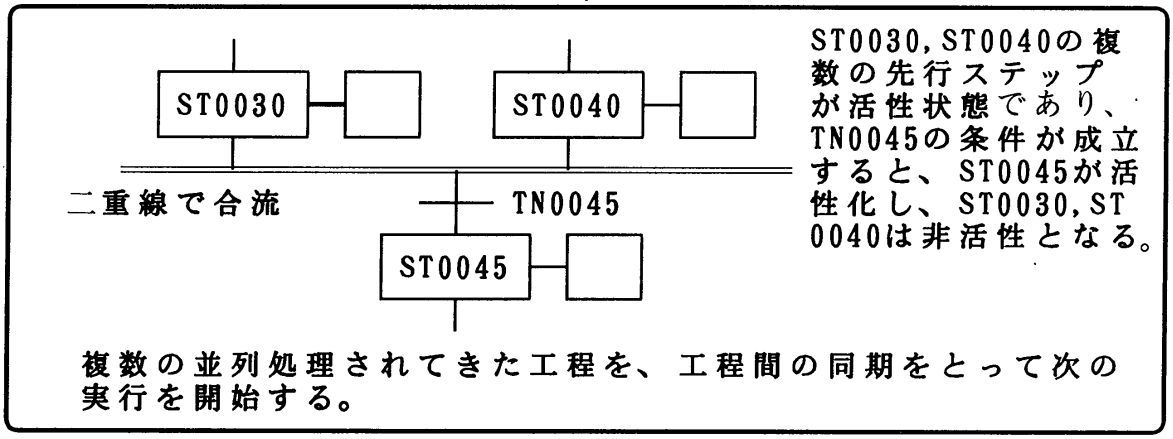


図5　並列合流

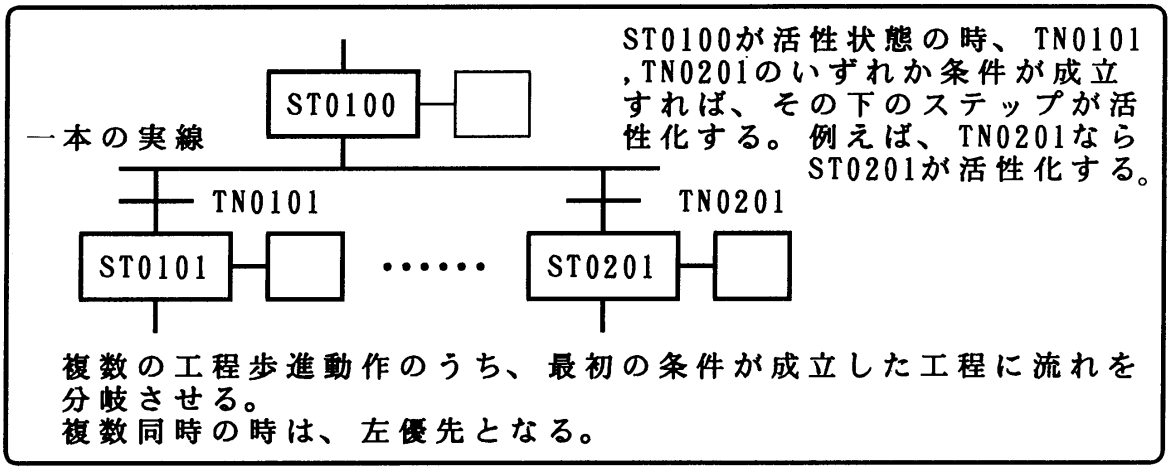


図6　選択分岐

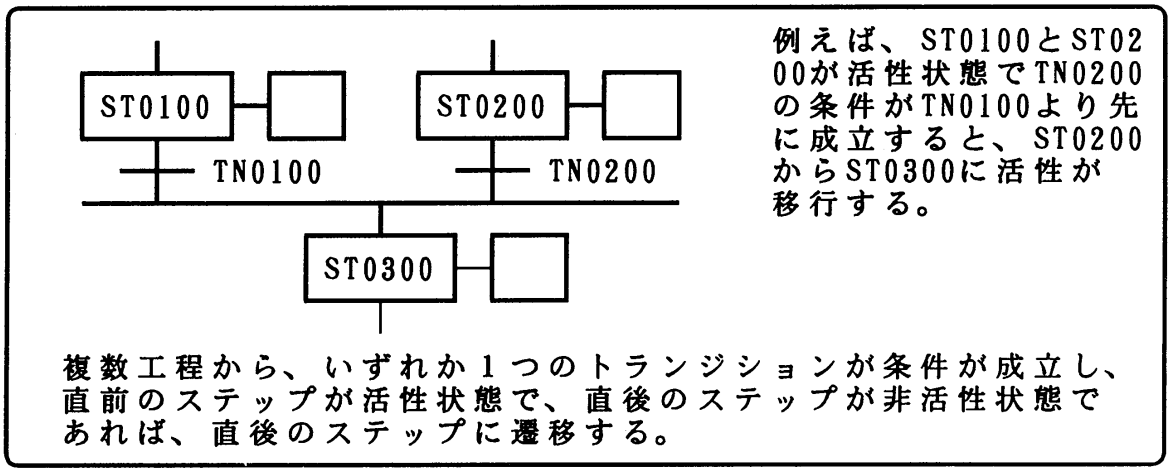


図7　選択合流

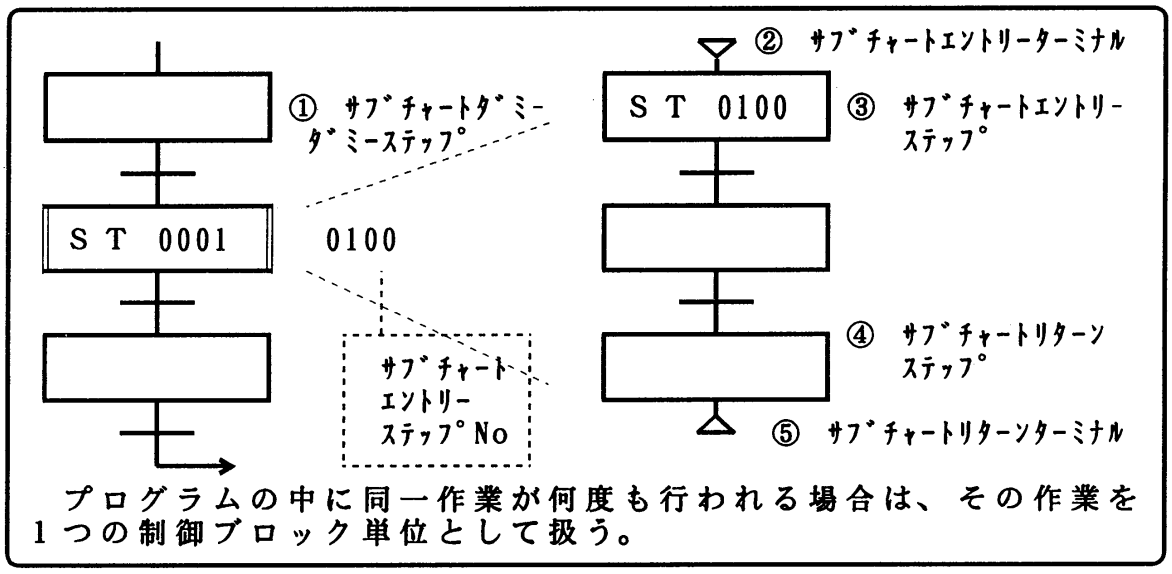


図 8　サブルーチン（サブチャート）

2-4．プログラムの構造化

計算機のソフトウェア設計では設計の容易さ（デバック、ミスの防止)、プログラムの再利用、プログラムの部品化、保守の容易さ等のメリットを考えたプログラムの構造化が実施されているが、シーケンス制御ではラダーを使用していることから構造化はできなかった。しかし、SFC はプログラムの構造化ができることからこの期待が大きい。

プログラムの構造化には、階層化、サブルーチン化（図 8 参照)、分割化等があるが、これができれば、プログラムの標準化、パッケージ化が可能となり、必要なプログラムを連結して一つのプログラムにでき、プログラムの生産性の効率化・向上を図ることができる。図 9 に構造化したSFCプログラムを示すが、図 9 中の(ａ)部は標準化したプログラム連結の例であり、(ｂ)部は階層化、(ｃ)部は分割化した例である。

このことは、SFC は、概念設計、機能設計、実行詳細設計へとトップダウン式に容易に構造化が実現可能であり、複数の人によってプログラムを分割して作成できることを意味する。さらに、セル工程単位でプログラムの標準化やパッケージ化をしておけばＩ／Ｏの変更のみでプログ

SFCの構造化(階層化、サブルーチン化、分割化)

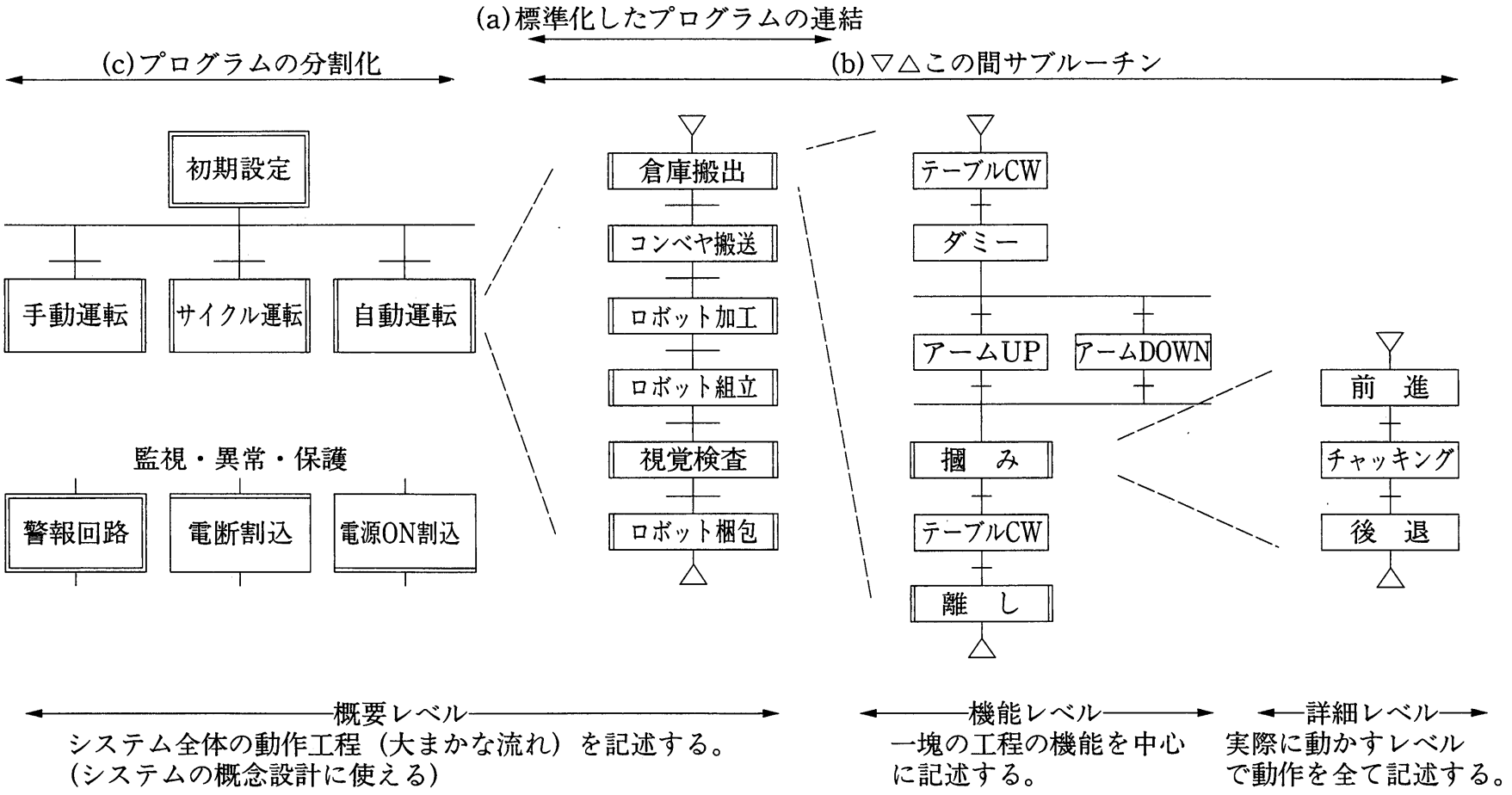


図 9　SFC の構造化

ラムの再利用ができる。また、ラダーは、シーケンスの中に手動・自動プログラムや異常・監視プログラムは埋もれてしまうが、SFCは別々のブロックとして扱うことができるので、設計が非常に容易となるし、見易くなる。このことはプログラムの改造やデバッグそしてメンテナンスの際に有効となる。

3．SFC問題点と技術課題

3-1．SFC普及の問題点（図11参照）

表2は、システム制御情報学会セミナー報告の資料である。ラダーを1とした時、SFCと比較したものである。（付録1は手法・実行方法・実行速度による比較である）

表2　（ラダーを1としたとき）

	ラダー	SFC
制御能力	1	1
文書量	1	0．3
教育工数	1	0．5
保全工数	1	0．5
試運転工数	1	0．3
設計工数	1	0．6～0．3

（システム制御情報学会　セミナー報告から）

このように、SFCは多くのメリットを持ち様々な期待をもたれながら今一つ普及していない。標準的なプログラミングスタイルとなって普及するためには、解決すべき問題点が幾つか存在すると考えられる。ここではPCを利用する際、ユーザがSFCを採用する時に発生するであろう懸念事項を採り上げてみた。

(1)　認識不足（誤解）・不安

設計者やその上司あるいは担当部門が、SFCに関する情報不足や未経験に伴う不安感からSFCに対する誤解、疑問、不信感等がある。

その一つは、SFCはプログラムの実行制御単位の要素であることか

ら、実行レベルプログラム（図10参照）ではラダー等のPC言語を併用しなければならず、SFC以前にラダー技術をマスターする必要があることから、やはり全てラダーで良いと言う思い込みがある。もう一つは、フローチャートの一種であるか、または、亜流と認識してフローチャートと同じように全てのアプリケーションをカバーできないものと誤解をしている。（付録2参照）

また、PCメーカごとにアピールの仕方（内容）が異なり、目的・利用方法が統一的に明確化されていないのも問題である。SFCは表現要素なのか言語なのか、ラダーを補完するものか置換するものか、等SFCのとらえ方が異なって認識されている節がある。

次に、将来性であるが、SFCを採用する側にとっては、全てのPCにSFCが搭載されていない現状下、全てのPCメーカ側が①将来にわたってSFCの製品の改善、技術サポート等を担ぎ続けて行くのか、②SFCが今後主流となり得るのか等の不安材料に対して明らかな姿勢表明がなければならない。一方、ラダーではすべての制御が実現でき、事足りているが、SFCでは、すべての制御に適用できないのではないかという不安もある。規格においても、1993年にIECに正式設定されたが、改良点もあったことから、JIS化が遅れ1995年末に工業技術院に提出されたように、規格内容の不明確さも不安原因のひとつとなっていることが考えられる。

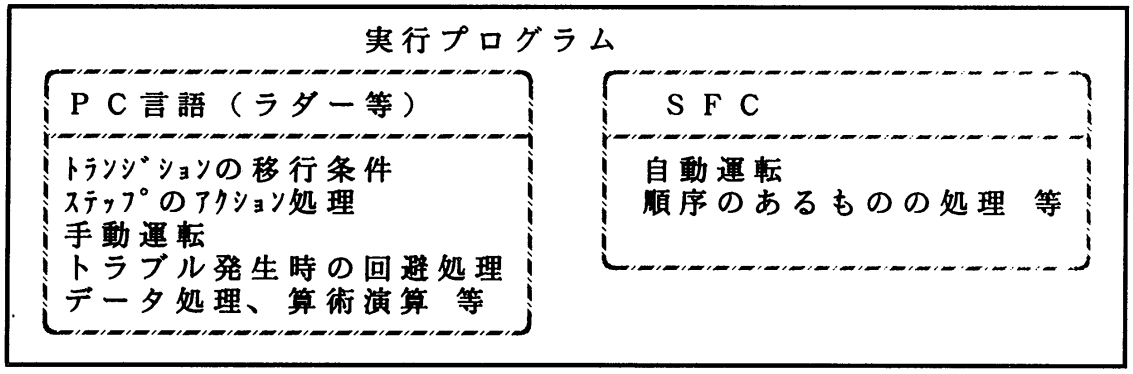


図10　SFC実行プログラムの構成

(2)　定量化しにくい・効果が見えにくい

SFC導入のメリットとして想定されるほとんどが、アプリケーションソフトウェアの生産性やメンテナンス作業の効率である。その効果が直接定量化（処理性能や通信容量）できる他の PC の機能と比較して定量化しにくい点がある。

また、試行や導入の際に、プログラミングツールが変わると“使い勝手”も変わり、設計プロセスの変更により立ち上がり工数（方法に慣れるまでの必要工数）が多くなることも効果の定量化を妨げているようである。

(3)　教育に必要となるパワー

一般的に新しい言語の採用はユーザにとって、それを習熟するための教育に関する投資や、過去から蓄積されているプログラム資源の扱いという観点で、かなりの決心が必要となる。採用言語の変更は、場合によっては設計プロセスや設計スタイルにまで影響する。

ラダーは状態と条件の組み合わせで処理を記述していくし、SFC は状態の遷移を基軸にして処理を記述していく一種の手続き型言語であるといえる。このような基本的なことによっても設計者・保全員に少なからず影響を及ぼす。ただし、これからの若い設計者への教育という観点では、数十キロステップという容量のラダーが組める技術者を育成することと、SFC を前提としたプログラミングスタイルを習得させることを比較した場合、コンピュータ的な知識を持った人材にとって後者が楽になるということは言うまでもないだろう。(SFC を用いてもラダーは必要になるが、数ステップのラダーを数十本組む能力と、数十キロステップのラダーを１本組む能力は比較できないほど格差がある)

また、理論上ではなく実際の設計業務において必要とされる様々なプログラミングに関するノウハウの不足を補完するための習熟期間および情報も必要となる。言い換えれば、SFC に関する解説書、プログラミン

グ例、教育セミナー（講習会）等教育体制が絶対的に不足していることがあげられる。

(4)　表記、動作仕様のばらつき

SFCに関する機能全般について各社ともIEC規格に準拠するとしているが、名称、表現、動作がメーカごとの製品によって部分的な差異がみられ完全には統一されていない。その実例を表記法の一部ではあるが表3に示す。また、SFCをコンパイラしてPCにローディングする方法が統一化されていない。

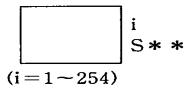
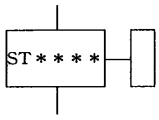
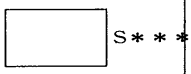
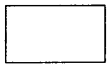
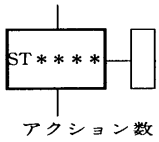
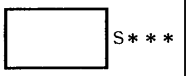
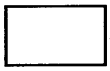
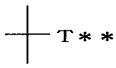
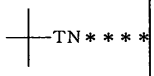
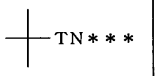
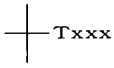
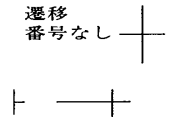
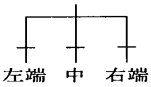
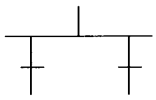
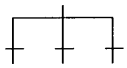
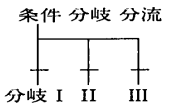
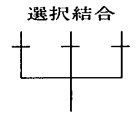
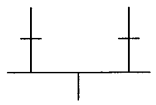
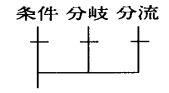
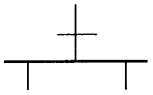
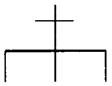
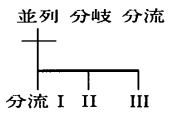
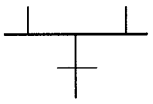
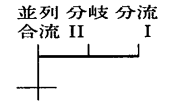
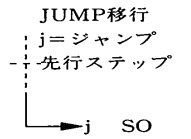
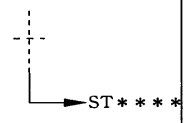
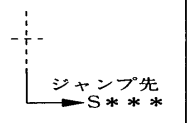
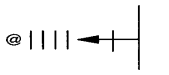
SFCと云っても自動運転／手動運転の切り替え、トラブル発生時の回避処理、データ処理、算術演算等制御対象の動きと直接関係のない処理はラダーで行うわけであるが、特に、自動運転／手動運転の切り替え、トラブル発生時の回避処理等をプログラムに組み込む場合、SFCで表現される工程歩進以外にその流れを制御する（活性状態を変更する）手段はIECやJIS規格には定義されていないため、各PCメーカでは用意しているが、残念ながら方法は統一されていない。

(5)　機能仕様面

SFCをそのまま現状のラダーの置き換えとしてとらえた場合や、SFC（スッテプ、トランジョンと接点を表すアクションのみ）ですべて表現してしまおうとした場合は、ラダーやその他の言語表現と比較して、①工程歩進としての表現が前提であること。②割り込み動作や演算等の動作ができない。等の不自由さがある。

SFCはラダーと同等の位置づけにあるものではなく、言語階層的にはラダーの上位に位置づけられるもの（ラダーで補完）と考えられるが、機能的にはON・OFF制御までは表現可能となっている。この点が使用者にラダーと同格のものという誤解を招いている原因である。したがって、このような認識が確実に存在しうることを課題として認識していなければならない。

表3　SFCの構成要素（各社一覧）

	A社	B社	C社	D社	E社
1 ステップ					
2 イニシャルステップ					
3 トランジション					
4 選択分岐					
5 選択合流					
6 並列分岐					
7 並列合流					
8 ジャンプ					
9 ジャンプエントリ					
10 エンド		なし		なし	

（職業能力開発大学校研修センター1994年3月刊行「教材情報資料No.19
新しいプログラム表現法(SFC)による生産自動化システム設計」から）

3-2．解決を必要とする技術課題

前に述べたSFCの普及に対しての問題点を踏まえて、主にPCメーカが解決して行わねばならない課題を挙げる。

(1)　SFC製品仕様の統一

まずは現有製品において統一されていないSFCに関わる名称、表現、動作等の整合が必要であり、社内・外，国内・外における標準化には不可欠な要素である。その際にベースとなるのはやはり規格となるだろうが、規格自体の解釈として不明瞭な部分や、記述として不足している部分が存在する場合は早急に補正される必要がある。その上で各PCメーカは、IEC 1131-3規格ベースにより製品を仕様化すべきと考える。この場合、現有製品の仕様との格差が問題となるが、問題点を解決しなければ、最終的な負担はPCの利用者におよび、トータルとしてSFCを搭載した製品の採用の阻害要因となってしまう。

(2)　アプリケーション研究

SFCは現状大半を占めるラダーユーザにとって、ある意味では全く新しい概念の記述手段である。その特長やSFCがもたらす効果、その文法を理解しただけでは、それを用いた設計をはじめとする実務での利用は難しいだろう。ましてハードウェアコスト等により多くの制限事項が生ずるようなアプリケーションプログラムの設計においては、文法等とは次元の違う様々なノウハウが必要とされる。これに対して上記に関する情報の事前提供が不可欠であると考えられる。

マクロ的には一般的なプログラムモデル（ガイドライン的なもの）の確立が必要であると考えられる。前にも述べたが、大半の設備機械に付帯される自動・手動運転等のモード切り替え、トラブル発生時の回避処理等を含めたプログラムモデルが必要である。このモデルは厳密にいえば設備機械の種類にあわせて形を変える可能性があるが、基本モデルの応用によりかなりの部分がカバーできるのではないだろうか。また、イ

ンタロックや自己保持等のミクロ的なプログラミング・テクニックもできれば欲しい情報である。

(3) SFC の検証方法の確立

SFCでプログラミングすることにより、従来のラダープログラミングでは発生しなかった活性状態でのデッドロックという問題が発生する。これは同時シーケンスの収束という活性状態の同期の表現により発生する問題であるが、静的な検証によっては回避できないケースが存在する。この例のようなSFCのプログラミング上の問題に対して、動的な検証につながるような技術（シュミレーション技術等）の確立が必要である。

(4) 技術以外の課題

前記に挙げた技術的な課題以外にも、以下の点についても対処することが必要である。

1)　教育訓練

SFC の正しい認識を妨げている原因の一つに、解説書、製品のマニュアル、セミナー等での説明不足がある。特に PC メーカが提供するものは、SFC一般についてあてはまることと、メーカが提供する製品固有のものの区別が曖昧になっている傾向がある。マニュアル・セミナー等は製品固有の仕様情報の提供手段でもあるため、ある程度の混在はやむを得ないが、製品固有の仕様と一般的な仕様を区別して伝えることが必要である。ましてSFCの位置づけ、捉え方レベルでメーカ間の格差は確実に避けるべきである。メーカ以外の教育機関においては、上記の問題についてはメーカ以上に留意しなければならないから、メーカ固有の仕様に偏れない。この様な中で独自のシステムを製作するとなると教育機材コストへ影響してくることから、各社製品の仕様統一が必要条件となる。

現在のところ仕様の統一がされていないので、公共施設（特に公共職業能力開発施設）で実施される訓練及び SFC 関連セミナーにおいても、使用する PC により若干なりとも使用テキスト、実習課題集にメーカ色

が出て来ることはやむを得ない。

2)　PR活動

規格化、製品化によりSFCの認知度はかなり上がってきている。ただし、前項で述べた問題も同時に発生しているようだ。製品を通してのSFCのPRは歓迎すべきことだが、製品PR以前にSFC自身がもつ狙いと効果、その理由を明確にしておくことが重要である。また、SFCが一過性のものでなく将来においても不変的であることを十分に伝えることも必要である。

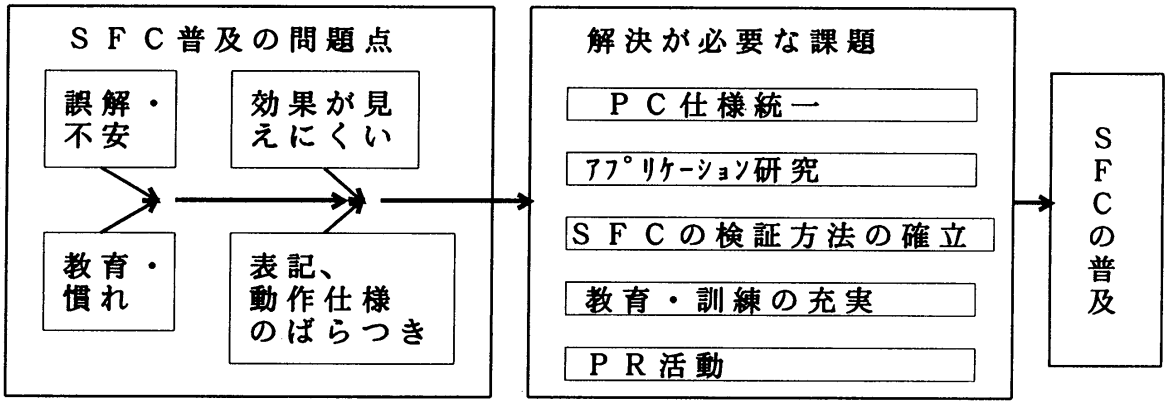


図11　SFC普及の問題点と解決が必要な課題

現在多くの企業は、ラダーに欠点はあるものの装置は正確に作動し、現状に満足しているものを、幾つかのリスクを負ってまで新しいSFCに変えるという冒険の必要性はなく、また、不景気のため設備投資を抑えていることからも、普及は難しいと考えられる。もしSFCの普及が促進されるとすれば、規格化の充実やSFCのPRは勿論だが、PCメーカ各社が全機種にSFCを標準搭載し、特に自動車産業のような大手エンドユーザが標準採用し、これに伴う機械メーカ、制御盤メーカの教育などが必要条件となる。また、コンピュータソフトウェア言語やパソコンソフト等に慣れ、ラダー教育を受けたことのない若い世代に交代すればSFCの普及は促進されるであろう。

これらの課題は、各PCメーカーや、SFCの普及に努めている

「SFC 研究会」等においても、徐々には解決されつつある。

3-3．SFC の普及と期待

現在の企業（特に設備機械）の進む道は、人手不足や製品の多様化等環境の変化を克服することにある。この場合、効率向上が大きな目的となり、CIM や FA が手段として取り入れられている。このことから、アプリケーションソフト容量の増大にともないプログラムの生産性向上のための設計効率・品質の向上等が重要視されている。また、これに伴い現場での保全の効率向上が不可欠となっている。設計効率が悪く保全性の悪いラダーに限界を認識し、生産性向上のために他の言語を考えている部門や企業では、SFC の導入の機運が非常に高くなっている。

(1)　数10キロステップのプログラムを数本～数十本も作らねばならないプラントのエンジニアリング部門や会社では、①運転方案（制御仕様）を SFC で記述、②１本のプログラムを複数の設計者で分担設計、③プログラムの階層化や標準化及び再利用、を考えてプログラムの生産性を向上するために SFC の導入を検討している。

(2)　設備を自社で設計・製作しているエンドユーザーが自ら設計・保全を可能とするため、プログラムの生産性向上を考えている設計部門及び現場での保全効率向上をめざす保全や整備部門は、導入を検討している。

(3)　ラダーで事足りている企業でも、設計者の経験やセンス、レベルによる設計品質や期間、経験の浅い設計者による設計期間の長期化の問題、現地調整期間の後ろ倒れの問題等が指摘され、その対策として SFC の導入を検討している。

(4)　国内・外問わずシステムを提供している大手企業で、国内用と海外用の標準開発言語の二重投資を強いられている部門等が積極的に取り入れようと検討している。

このように PC を多数使っている大手のエンドユーザで、既に SFC の

採用に至っているPCユーザでは、具体的には以下のような項目の評価があがっている。また、採用を検討しているPCユーザでも、ほぼこれらと同様の効果を期待しているようである。

3-3-1．SFCに期待するところ（図12参照）

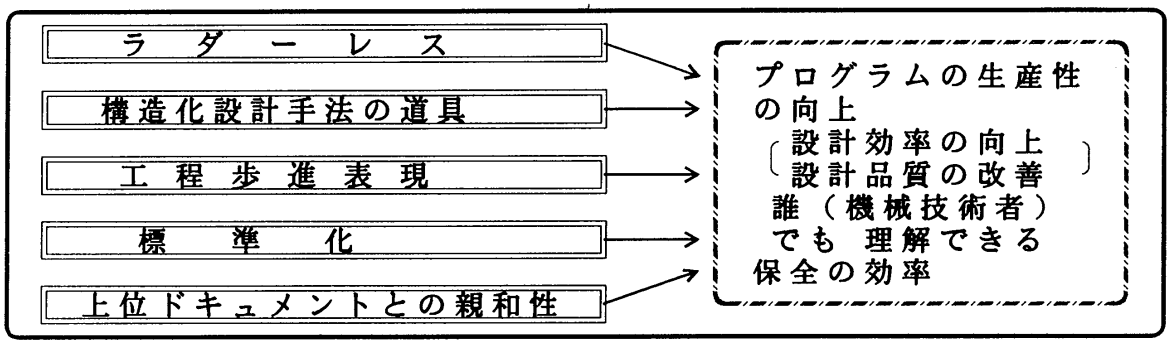


図12　SFCに期待するところ

(1)　ラダーレス

PCのアプリケーションソフトを記述するラダーは、圧倒的に主流であるが、この言語は、リレーシーケンス図をプログラム言語化したものであり、「入力と出力の関係がわかりやすい」「論理条件が視覚的に認識できる」「条件のさかのぼりが出来る」等、多くの長所がある。また、PC設計者により物理的にも、ノウハウ的にも永年培われてきたソフトウェア資産も豊富であり、その支持率、利用率は非常に高い。しかし、アプリケーションプログラムの容量の急激な増加に伴い、局所的な視認性より全体的な視認性がより重要となり、その設計技法においても個人的な能力に依存し切れなくなりつつある。一般的にラダーの短所とは、①視認性の悪さ（スパゲティ型のプログラム構造になりやすい）、②個人スキルの依存性が高い（他人が作ったプログラムが理解しにくいし、メンテナンス性が悪い）、③プログラミングノウハウの構築に時間がかかる。等挙げられている。このことは、ソフトウェアの生産性及び品質の向上を阻害していることにもなる。

SFC は視認性の高さとそのカリキュラムにおいてもコンピュータ的な知識は益々その比重を大きくしている。欧米では陳腐化しつつあるラダーに拘るより、コンピュータ的な知識でプラグラミングできるほうがPC ユーザでの教育工数削減ができるのは明らかであるとして普及が進んでいる。

(2)　構造化設計手法の道具

現在、ソフトウェア設計においてかなり実績のある構造化設計手法をSFC に利用することにより、PC のアプリケーションプログラムの設計プロセスの効率・品質の向上が出来るのではないかという期待がある。

一般的に、ソフトウェアの生産性および品質を向上させるためには、ソフトウェア工学に基づいた設計技法を取り込むのが効果的である。しかしながら、ラダーは一般的に普及しているＣ言語（高級言語）やアセンブラ（低級言語）などと比較してもかなり特殊な言語であり、それゆえにこの分野への構造化設計手法の展開は、技術面からも環境面からも困難であった。

SFC によるプログラミングを前提に設計するとコーディング以前に最適なプログラム構造が確立できる。また、ラダーと比較して階層化、モジュール化した結果が明示的に現れるため、設計効率、メンテナンス効率、品質の向上に有効である。

(3)　工程歩進表現

ラダーは基本的には、状態と条件の組み合わせにより出力をどうするかをプログラミングしていく言語である。そのため時間軸をもった動き（動作・工程の流れ）は表現しにくい。SFC は工程歩進的な表現が基本となっているため、時間軸をもった動作が表現しやすいし、図示表現であるがゆえに時間軸による工程の流れが形に現れる。したがって、SFC がラダーの苦手な時間軸をもった動きに対する表現をとらえるのではないかという期待がある。

(4)　標準化

ソフトウェアの生産性や品質の向上をさせるための考え方の一つとして標準化がある。ここでいう標準化には、大きく2つの意味がある。

一つは、会社・職場等ある限られた範囲において、ソフトウェアの構造またはソフトウェアの一部を再利用可能な形に整え、共有するという意味での標準化である。特にSFCは、①プログラム構造をパッケージ（パターン）化できること、②アクションを用いて明示的に再利用部品の構築ができること、等の特徴がある。

もう一つは、PC全体の製品仕様の標準化についてである。現状では、ラダーについても大枠の文法は共通となっているものの、その詳細に至るとメーカが他社製品に差別化できる製品機能を搭載している。これはユーザに対して付加価値を提供している反面、複数メーカ製品を利用するユーザには、その仕様の違いを認識かつ理解せねばならないという負担が発生している。そのため仕様の標準化を求めるユーザの声が国内・外ともに強くなってきている。これに応えるには、ラダーの標準化はPCメーカにとっての負担も大きく、ユーザ側にも標準化された仕様をマスターせねばならないという負担が生じる。しかし、SFCは、ラダーよりも製品化の歴史も浅く、また、製品化に先行して規格化が行われたため、現状メーカ間の仕様の差異が少なく、ユーザ側でも新規にマスターしていく段階であるため、現実レベルでの標準化の可能性が高いと考えられる。この2点からSFCがその標準化に有効な道具になるのではないかとの期待がある。

(5)　上位ドキュメントとの親和性

SFCは機械をどのように制御するかを記述する表現法であることから、PCの機種に依存する面は少なく、同一制御対象を異なる機種のPCで制御する必要があるとき、SFC部分の記述はほぼ同一になる。このことから、設備機械の動作フローを表現した運転方案からPCのプロ

グラミングが容易に行われることへの期待がある。また、これは将来的には運転方案から PC のアプリケーションプログラムが自動生成されるような状況も充分に予想されることを意味する。

3-3-2．SFC のアーキティクチャ

前項のようなPCユーザの期待を受けて、各PCメーカではどのような対応をしているのだろうか。採り上げるべき機能やツールは数々あるが、今回はその基本的な構造、つまり、設計者によりプログラミングされた SFC がいかにしてその実行をつかさどるユニットに送られ処理されているかに注目して、SFCのエディターおよびコンパイラを搭載したプログラムローダーから PC ユニットに SFC をローディングする方法として、現在下記の 3 方式が採用されている。厳密にみれば各メーカの方式には違いがあるが、おおまかにはこれらの方法のいずれかに当てはまると考えられる。以下に、そのアーキテクチャと方法のメリット・デメリットを紹介する。

(1)　ラダー変換型

図示表現により記述されたSFCを周辺機器でラダーに変換し、コントローラ側ではラダーを実行処理することにより SFC 動作を実現する方法である。国際的には、PC のプログラミング用 CAD に普及しているが、日本ではA社がこの方法を採用している。(図13)

この方法のメリットをユーザー側とメーカ側に分けて考えると、

ユーザ側のメリットとしては、最終的にラダーに変換されるためラダーに慣れ親しんだ作業者にも違和感がない。なんと言っても現場でラダーに変換できることにある。そして、これが単一 PC メーカーの専用ローダー（コンパイラ）ではなく、複数PCメーカーのラダーオブジェクトへの変換を実現しているものであれば、他メーカ PC への互換性が高くなり、SFCを搭載していない機種でもSFCプログラムを実行できるメ

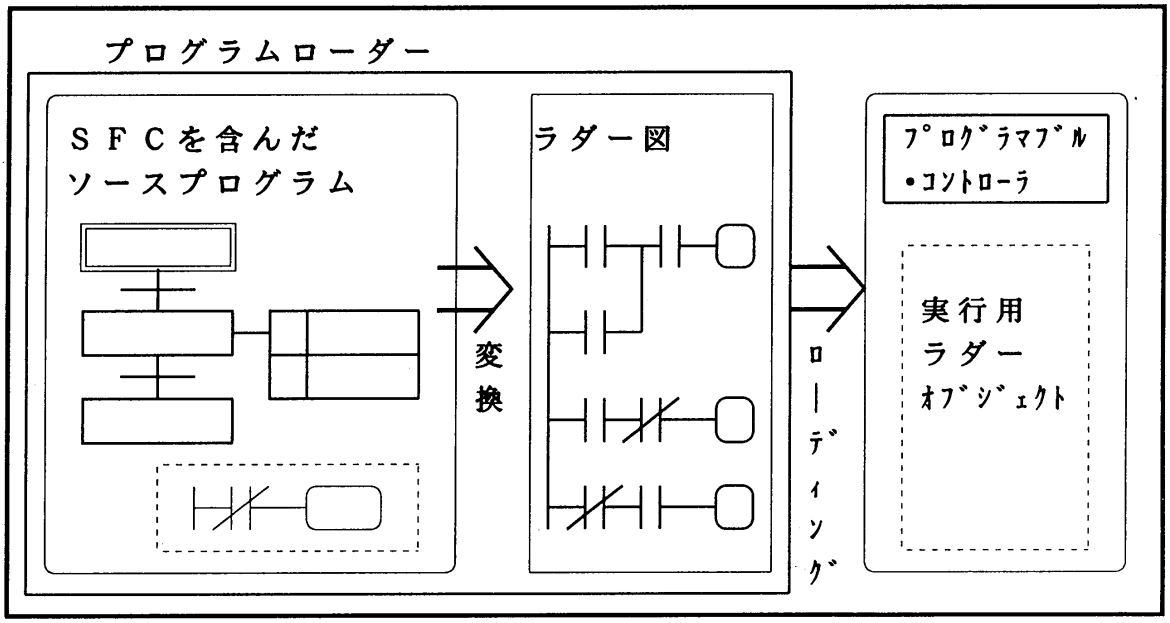


図13　ラダー変換型

リットがある。

メーカ側のメリットとしては、ローダーだけの開発で現有のラダーでSFCが実行できる。デメリットとしては、ラダーレベルの変更に対して図示表現への逆コンパイラ（逆変換）が困難であるし、オブジェクト容量、実効速度に冗長性が発生する。

(2)　SFC 独立型

ラダーを論理的（もしくは物理的）に分割し、個々の実行タイミングを制御するロジックとしてSFC部をラダーから独立させる方法で、ラダーとSFC両方のプログラムが実行できる。(SFCプログラムは専用処理をし、ラダープログラムにコンパイルしていない。)

この方法は国内では主流でる。(図14)

メリットとしては、オブジェクトレベルでのプログラム変更に対して、図示表現への逆変換が可能（容易）であり、オブジェクト容量、実効速度に冗長性が少ない。デメリットとしては、ユーザ側から見ると最終的にラダー表現に置き換わるだけではないので、PCのプログラムに関わるすべての作業者がSFCを習得することが必要になる。メーカ側では、

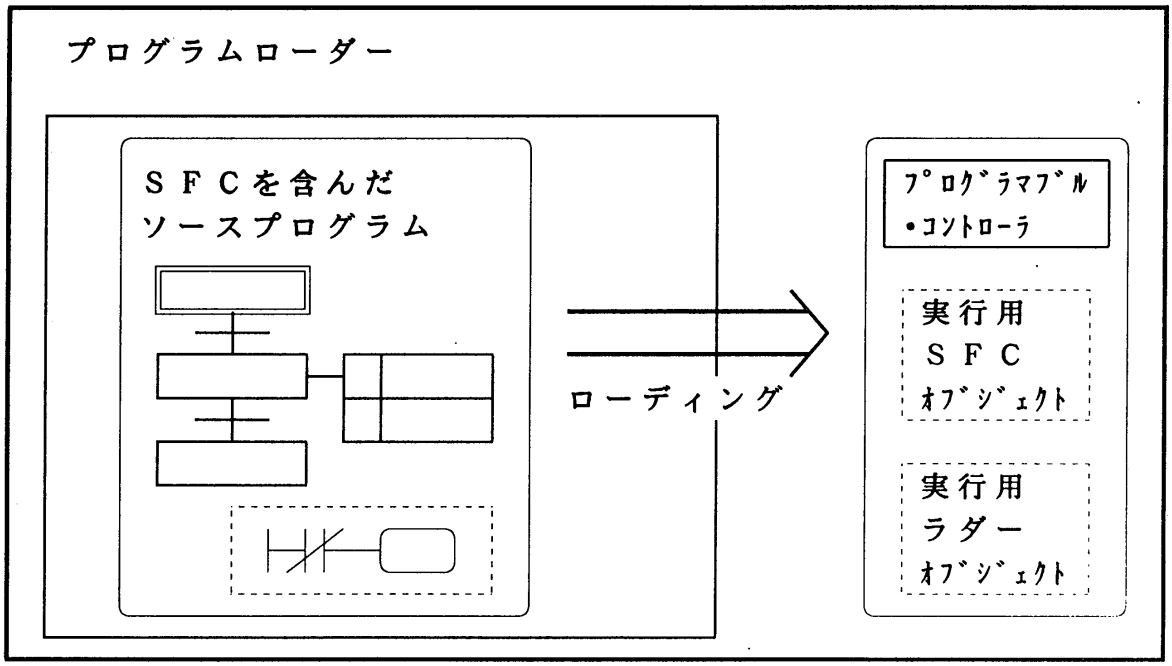


図14　SFC 独立型

ラダー以外に SFC で表現されているロジックを実行する特殊な処理が必要となる。

(3)　一括変換型

SFC、そしてその中に組み込まれるラダー、その他の言語表現を一括して他の言語もしくはインプリメントされるコントローラ側が解釈できる機械語に変換し、コントローラ側では機械語の連続したプログラムロジックとして処理する方法である。(図15)

メリットは、パソコンなどの標準プラットフォームに対して組み込みやすい (利用しやすい)。デメリットは、オブジェクトレベルでのプログラム変更に対して図示表現への逆変換がかなり困難であり、順序のない制御が記述しにくい。ラダー変換型以上にオブジェクト容量、実効速度に冗長性が発生する。

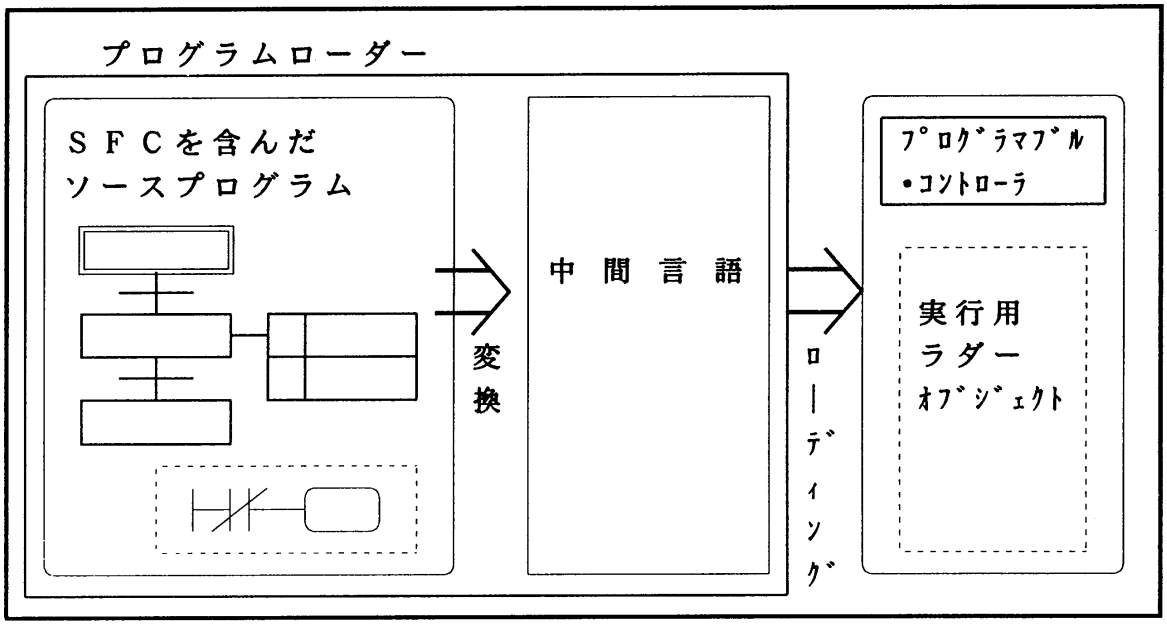


図15　一括変換型

4．SFCの今後

PCユーザのアプリケーションプログラムの生産性向上に対する取り組みを支援するものとして、SFCは既にキーワード化しているが、これまで述べてきた課題やPCメーカの取り組み状況を踏まえて、SFCの今後の方向性を予想してみる。

4-1．2局化

現状、不明確だったSFCの位置づけと利用対象が明確化されることにより、SFCの役割が大きく「効率化ツールとしてのSFC」と「簡易工程歩進表現としてのSFC」の2つに分かれていくことが予想される。

前者として、設計効率の向上を余儀なくされる規模のアプリケーションには、開発技法や効率化ツールとしての側面がより一層要求されてくる。冒頭で述べたようにPCのアプリケーションとしての全体比率もこちらが高くなってくるであろう。SFCという言語要素を用いた構造化設計や上位ドキュメントとのリンクという設計ツールとしての側面と、設

備・機械の動きをグラフィカルに表現しているという特徴を生かした運用ツールとしての側面の両面での発展が期待できる。（A社、B社）

後者としては、プログラムの視認性や技法による設計効率を必要としない比較的小規模のアプリケーションプログラムにとっては、あえてプログラムを構造化する必要はなく、むしろ工程歩進表現できる言語的な側面がメリットとなるため、より簡便さ簡易さの追求が進むであろうと思われる。（A社）

(1)　設計ツールとしての側面

「効率化ツールとしてのSFC」の発展は、「設計ツールとしての側面」と「運用ツールとしての側面」の両側面での発展が考えられる。これをもう少し具体的にとらえてみると、「設計ツールとしての側面」では、構造化設計手法の取り込みという観点だけでなく現状のCASEツールで支援されているような環境が想定される。例えば、以下のような展開が考えられる。

・シミュレーション

ペトリネットで研究されているような、システムのふるまいの予想や、PC にダウンロードされる以前の動作シミュレーション。

・アクションのライブラリ化

C言語等の高級言語に用意されているような標準関数的なアクションの提供。

・プログラムの自動化

一定の記述ルールにより定義された要求仕様書、運転法案からのプログラムの自動生成。

(2)　運用ツールとしての側面

SFC はプログラム自身が機械・設備の動作をグラフィカルに表現する手段としても利用できる。既に製品化されているものにも、チャート上でステップの活性状態をリアルタイムに反映させることにより、設計

対象の設備・機械の動作状況をモニタリングできるデバック機能は既に搭載されている。現状では圧倒的に定着しているラダーを用いた方法に慣れた保全員にとって、SFCを用いた方法はやや馴染みづらい状況のようである。今後は対象の動作を記述しているという特徴と、視覚的な特徴を利用した様々な機能サポートと相まって確実に浸透していくであろうことが予想される。その「運用・メンテナンスツール」としての側面において予想できる具体的展開には、例えば以下のようなものが考えられる。

・監視系ソフトウェアとのリンク

現在、注目されているWindowsベースの監視ソフト等のモニタリング画面として、メンテナンス機能の一部としてアプリケーションプログラムの実行状況（SFCの活性状態）をよりグラフィカルにした形でインプリメントしたもの。

・トラブルシューティング

設備・機械のトラブルに対して、どのような状態で、かつアプリケーションプログラム上のどの部分が問題があるかを自動的にガイダンスする機能（SFCの活性状態がそのまま動作状況につながるまでこの特徴を利用）また、各工程間の遷移時間をSFCの活性状態の遷移時間として計測し、基準値を越えた場合にその異常箇所を併せて警告するような機能。

4－2．進む周辺機器の取り込み

もう一つの方向性として、SFCがシーケンス制御だけでなく、それ以外の制御（サーボ、情報処理、プロセス）を包含するシステム全体の状態記述に発展することが考えられる。

(1)　シーケンス制御とそれ以外の制御との関係

PC（特にラダーを処理するユニット）は、I／OのON・OFF制御だ

けでなく、システム全体の排他制御やタイミング制御（たとえば個々のサーボモータ制御に対する統括的な起動・停止）を行う役割がある。

SFC は、これらの制御は苦手とされていたが、現在ではその大部分において可能になっている。将来的にはSFCにより周辺機器、コントローラの動作状態まで記述可能になるのではないだろうか。

(2)　マルチ言語化

システム全体の運転方案からプログラミングまではSFC で表現され、個々の制御はアクションとして組み込まれるようになるとそれぞれの制御に適した言語表現でアクションが記述できることが必要条件となる。

現状、IEC 1131-3では、アクションとして組み込まれる言語として、図示言語ではラダーとFBD、テキスト言語ではILとSTの 4 言語が定められている。ここでいうマルチ言語化とは、これらの言語バリエーションが増えていくことにより実現されるものである。

(3)　国際標準として確立

日本のインダストリアル・コントロール・システムは、現在、そのほとんどが専用システムになっており、ユーザはもちろんベンダー自信も窮屈な状況に陥っている。例えば、A社の専用開発言語が他社のコントローラや海外市場で使えるかと言うと NO である。海外市場でのインターオペラビリティを実現するため、日本市場向けの専用開発言語と国際市場向けの標準開発言語の二重投資を強いられているところがある。そして、日本のユーザは、ベンダーごとに異なる開発言語を前に、ユーザの志向にあったシステム導入がなかなか実現しないという不便が続くことになる。この様な背景は日本だけの問題ではなく世界各国にもみられる。SFC は主に欧州で規格化されてきた経緯から、その普及も欧州が中心であった。PLCOpen と言う PC メーカ，PC ユーザを交えた国際標準化組織が1992年オランダで設立され、PC 言語の国際標準規格IEC1131-3の普及を通じて、オープン化に取り組み間接的にSFCの普及

をバックアップしている。米国においては、欧州と比較してSFCの普及は進んでいなかったが、米国自動車ビック3が共同で定めた機器導入指針のOMACペーパには明確にIEC 1131-3規格の互換製品によるプログラミングが規定してオープン化を唱えている。また、OSACAは欧州の工作機械やPCメーカそれに研究調査機関が一体となって、国際競争力を改善するために設立された欧州共同プロジェクトで、主要目的は企業色のないオープンコントロールシステムのアーキテクチャを定義するとしてオープン化を進めている。この様に、オープン化の流れは、今後世界を含めて主流になっていくことは間違いないと思われる。

また、これを契機に自動車産業以外においても、IEC 1131-3の位置づけが上がってくることが予想される。

上記の欧米の動きと国内産業の海外進出が進んでいる昨今の状況を考えると、国際的な規格として定着しつつあるIEC1131-3への製品仕様の準拠の加速度は更に強まることが予想できる。それに伴いSFCの普及度も高まって行くであろう。ちなみに、国内規格においてもJISの国際化の一環としてIEC 1131-3のJIS化が完了しつつある。

5．終わりに

SFCについて、“SFCとは”に始まって、“SFCの今後”と述べてきたがこれをまとめると、21世紀が目前に迫った現在、わが国は、大きな変革の流れのまっただ中にあり、長引く景気低迷のもとで、製造業を取り巻く環境は、円高など依然として厳しい状況にある。しかし、製造業は、市場動向に迅速に柔軟に対応し、国際的に競争力のある高品質・高付加価値製品の研究・開発が迫られている。一方PCは、生産現場のFA化に伴い生産システムの柔軟性、拡張性や装置の信頼性の追求、操作性の向上など製造工程の簡素化・合理化に利用されている。

PCプログラミング要素であるSFCの長所は、ラダーよりもプログラ

ムの生産性向上の良さと、メンテナンスに優れている点である。そして、IEC規格をもとにSFCをPCの世界共通語とし標準化・オープン化する動きが欧米中心にあり、①　PLCOpen、②　OMACペーパ、③　OSAC等の国際任意団体がその例である。特にヨーロッパ（仏）では制御の45%以上の高使用率である。日本でもJIS化はIEC規格に準拠して進められているし、SFC研究会なる団体で普及を唱えていることから将来性は高い。したがって、シーケンス制御の動きは、ラダーからSFCに移行して行くことは間違いないだろう。そのためには、SFC教育と普及活動が非常に重要であり不可欠である。

この様な動向の中で、職業能力開発大学校（以下、能開大）研修研究センター開発研究部においては、1992（平成4）年から「生産自動化システム制御に関する向上訓練コース開発　－新しい表現法（SFC）による制御コース－」を埼玉職業能力開発促進センターと共同開発をはじめ、1994年にはセミナー用テキストとその実習機が完成し、基礎編と設計編の能力開発セミナーを実施している。能開大研修研究センター研修課程部短期実践セミナーでは、SFCのコースを平成3年から開始し、現在は基礎編と応用編の二コースを実施している。そのためか、全国の公共職業訓練施設でも「SFCセミナー」が急増している。（平成5年度は一桁，平成6年度は16コース，平成7年度においてはポリテクセンター62コース，短大18コース，計80コース）また、1995年からスタートした雇用促進事業団の「人材高度化研修」の中にも「SFCセミナー」が組み込まれている。PCはFAの中核機器として幅広く活用されており、ラダーからSFCに移行する時期にある日本産業の現状から見れば、これらのセミナーの実施は、国内的にも国際的にも（小さくは海外センター協力、大きくは海外進出企業）将来性を見通したSFCの普及している形となっている。

参考資料

(1)　PLCOpen

PLCOpen は、IEC 1131-3の普及を通じて、プログラマブルコントローラのユーザに大きなメリットをもたらすことをめざす国際組織で、1992年オランダで設立され、極東の拠点として東京にオフイスを構えており、日本企業11社が会員になってユーザにオープン化のメリットとして次のようなものを提供している。

1)　山武社ハネウェル㈱FA日本事業部はIEC 1131-3ベースに、シーケンス制御とPID制御を融合したFAコントローラを開発した。ツールはWindows環境下での扱い易い画面と操作性、LD, FBD, SFCのマルチプログラム言語を採用しプログラム資産の有効活用、構造化、機能単位プログラムの再利用等が出来る。

2)　コマツ㈱エレクトロニクス事業本部は米国ビッグ3提唱のOMAC要求仕様でもある5種類のプログラミング言語IEC 1131-3完全準拠の「ISAGRAF」というPCパソコンソフトを提供している。

3)　ディジタル㈱はOMAC, OSACA等のオープンコントローラシステムとして使用でき、世界標準のPC／ATをベースに大形TFTとアナログタッチパネルによるパネルコンピュータ（OSは、DOS／V、WINDOWS、OS／2に対応し、市販の各種アプリケーションが使用可能）を開発提供している。

PLCOpen活動方針としては、次のようなことを上げている。

1　規格団体ではないが、IEC 1131-3を推奨している。

2　会員は、IEC 1131-3互換システムを提供しているか、利用することに共通の関心を持っている。

3　PLCOpenの環境を広くプロモートしていく。

4　共通するインプリメンテーション・コンセプトを開発していく。

5　製品の互換テストをおこなう研究機関を指定する。

(2) OMAC

米国自動車業界ビック 3 である GM, Ford, Crysler 社が今後の機器導入指針としてまとめたものである。背景には、日本や欧州諸国から自動車の大量輸入によって、米自動車産業界が設備機器のリストラを更に強固に実行するために、OMAC の必要条件をサプライヤや技術者グループに知らせるために作成したのものである。その必要条件とはアプリケーション上とアーキテクチャ上の 2 つのニーズである。

アプリケーション・ニーズとしては、安全性と信頼性、コスト、柔軟性、接続性、メンテナンス、トレーニングと言った項目をあげている。これは、現在自動車業界で使用されている PC は、サプライヤが持つ独自のコントロール技術からなっているため、ベンダー指導形の価格設定が行われ、機器には共通するインターフェイスが無く、インテグレーションコストが高く付き、かつ、トラブルや新規オペレーションの都度、特別訓練が必要となる等山積みする問題を解決したいということになる。そして、これらのニーズを満たすために、エコノミカル（ライフサイクルコスト）でメンテナブル（操作、迅速な修理、簡単なメンテ）でモジュラー型（コンポーネントの Plug and Play）のスケーラブル（簡単で効果的なリコンフィグレーション）な PC を導入する方針を打ち出している。アーキテクチャ・ニーズとしては 7 つの項目があり、①インフラストラクチャ，②インフォメーション・ベース・マネジメント，③タスク・コーディネーション④ヒューマン・インタフェイス，⑤モーション・コントローラ，⑥ディスクリート・イベント・コントローラ，⑦センシング・インタフェース等を上げている。なお、③，⑥に、明確に IEC-1131-3規格の互換製品によるプログラミングが規定されている。

(3) OSACA

欧州の工作機械（INDEX, HURON, COMAU）や PC メーカ

（ATEK, BOSCH, NUM, FAGOR, SIEMENS）の国際競争力を改善するために研究調査機関（WZL, FISW, CBTMittellan）も加わって設立された欧州共同プロジェクト（OSACA＝Open System Architecture for Controls within Automation Systems）のことで、主要目的は企業色のないオープンコントロールシステムのアーキテクチャを定義することにある。

CNCユーザの第1の目的は、CNC用の欧州標準を設定して、この分野の投資を減らすことであり、第2の目的は、メカ屋がメカ屋のために設計したソフトウェアのインプリメンテーションを可能にするためCNCをオープン化することである。

工作機械のメーカの目的は、工作機械メーカの意図でCNCを機械の統合された部品とすることにあり、機械を制御に適合させるのではなく、制御を機械に適合させることにある。そのため、OSACAは、全ての制御に対して統一したコンフィグレーション・メカニズムをもたらすこととしている。

(4)　SFC研究会

この研究委員会はSFCに関する研究を行い、問題点を明らかにしSFC技術のより高度な実用化とより広範囲な普及を図ることを目的とし、ユーザ、メーカ、学術研究者等が集まってSFCを調査、研究を行っている任意団体で、今後、電気学会の第2種共同研究委員会として活動する予定である。現在、SFCの問題点（QandA）の抽出や規格の研究を行っており、普及活動も始めている。

調査・研究事項として次のようなことを上げている。

1)　PCの実用技術に関する調査・研究

2)　SFCの実用技術に関する調査・研究及び普及

3)　本会の活動により得られた成果の普及

4)　その他本会の目的を達成するために必要な調査・研究

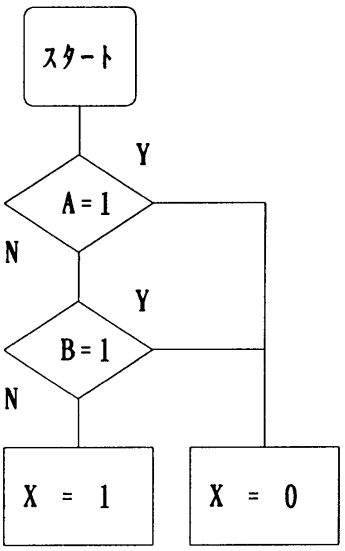
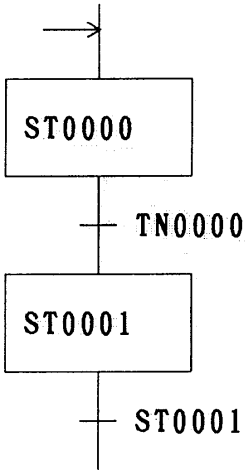
付録1

ラダー方式とSFC方式の比較

	ラダー方式	ＳＦＣ方式
手法〔実行方法〕	(ﾎﾞﾄﾑｱｯﾌﾟﾌﾟﾛｸﾞﾗﾐﾝｸﾞ) ・プログラム全体ができるまで繰り返えす。 ・それぞれの動作を矛盾なく関係付ける。 ・細部の動作から決める。 ・評価の条件は、常に全ての条件について行われる。 ・全ての部分が順番に実行して行くのでその時の処理とは無関係な工程への影響も考えなければならず、動作条件が複雑になる。	(ﾄｯﾌﾟﾀﾞｳﾝﾌﾟﾛｸﾞﾗﾐﾝｸﾞ) ・プログラム全体の流れを決める。 ・流れの個々の部分を明確にする。 ・細部の動作が決まるまで繰返す。 ・条件の評価は活性状態のｽﾃｯﾌﾟ及び移行条件による ・工程間の影響を考えなくても済む。
実行速度	・常にプログラム全体を実行している為、実行の１サイクルが長く、プログラム全体の実行速度が遅くなる。	・ステップ単位で演算処理するので全体的ﾀｸﾄﾀｲﾑが短くなり、早い処理や迅速な対応ができる。
表現記述性	・信号の組合せ論理を回路図表現。 ・制御順序の表現は難しい。	・制御の順序を遷移図で表現。 ・順序のない論理組合せだけの処理には向かない。
保守性解読性	・整理されたプログラムとコメントがないと、他人が理解できない。 ・制御動作をプログラムから読むのは困難。	・順序の流れ図で記述してあるので、分かり易く、保守が容易。 ・制御動作の順序はプログラムより読める。
言語	・ＰＣ言語	・ＰＣ言語の要素として

付録2

SFCフローチャート言語の比較

フ ロ ー 方 式	S F C 方 式
・流れが直線的なので、プログラムを論理的に見ていく必要がある。	・プログラムが工程単位にまとめているから、見ただけで工程の流れが大体理解できる。また、プログラムを構造化し易い。
・ＰＣ内での処理内容の手順を記述したもので、プログラムそのものである。	・ＰＣ内の演算処理の手順を全て示したものでなく、プログラムの実行制御単位の要素である。
	

共通点：工程全体の流れを追いながらプログラミングするトップダウンプログラミング方式という点

付録 3

SFC の歴史

１９７７年	ペトリネットの概念をＰＣプログラミング言語に応用した"GRAFCET"（ｸﾞﾗﾌｾ）がフランスより提唱された。
１９７９年	ＩＥＣでＰＣの規格化委員会（Ｗ６）が結成し、標準化作業がスタートした。
１９８８年	Ｗ６から "GRAFCET" を拡張したＳＦＣを含むＰＣプログラミング言語の規格案が提出された。（ IEC848 となる）
１９９０年以降	ＩＥＣ規格案をベースとしたＰＣプログラミング言語のＪＩＳ化が進んでいる。
１９９３年３月	IEC1131-3として正式に制定された。
１９９６年	IEC1131-3の翻訳盤としてJIS-B3503として制定 (IEC1131-3の改定があったことから1995年末に工業技術院に提出)

(ほんだ　まさお　職業能力開発大学校 研修研究センター 開発研究部)