U.D.C. 〔681.327.13.06:681.32.06.002:658.5.012〕
:681.322-181.48.06

マイクロコンピュータソフトウェア分野での“SEWB”適用

Applying “SEWB” to Micro Computer Software Development

増井光幸*　*Mitsuyuki Masui*
前澤裕行**　*Hiroyuki Maezawa*
広瀬　弘***　*Hiroshi Hirose*
山口文夫****　*Fumio Yamaguchi*

マイクロコンピュータを組み込んだシステム製品は，マイクロコンピュータの適用領域の拡大や市場ニーズの多様化に伴い，機能を実現するためのソフトウェア開発量が急増しており，ソフトウェアの生産性と品質の向上が急務となっている。この解決のため生産環境を整える必要があり，作業工数の負担が大きい中流工程以降の対応策として，ビジュアルプログラミング手法を取り入れた日立製作所SEWBを適用し，中流工程でのプログラム半自動生成を実現した。更に，通信システム向けに仕様記述法の一つである状態遷移図を支援する機能を実現した。以上のシステムを，各種マイクロコンピュータを組み込んだシステム製品に適用し，ソフトウェア開発の品質及び効率向上を図った。

1 緒　　言

各種システム製品は，機能の高度化や高性能化のために多くのマイクロコンピュータが使われている。

これは，16ビットや32ビット化などの高性能化，メモリの経済化などによって，ソフトウェアを用いて実現するほうがより有利となるため急速に普及していると考えられる。

以上のようなことから，マイクロコンピュータの適用範囲も拡大し，組み込まれるソフトウェアも高度化，大規模化し年々増加の一途をたどっている。

しかし，すべてのソフトウェアの開発需要に対し技術者の不足が叫ばれている状況にあり，マイクロコンピュータ用ソフトウェアも同様である。

これらの解決にはソフトウェアの生産性向上が急務であり，今日まで各種のソフトウェア開発を支援するツールが開発されてきた[1)~5)]。しかし，従来のソフトウェア開発を支援するツールの稼動環境は，

(1)　マイクロコンピュータ用開発専用機では処理能力の面で問題があり，大規模なソフトウェアの開発には向かない。そこで，はん(汎)用コンピュータを使用するクロス形が求められた。

(2)　はん用コンピュータによるクロス形ツールでは使用コンピュータの負荷率に応じて，端末の応答性の低下やジョブのスループットが低下する。

などの問題がある。

一方，利用面でもプログラム論理を分かりやすい形の図形表現をしても，その後，図形修正，コーディング及び端末から入力と，単純な変換作業を人手によっているため，この間の工数や品質の面でロスがあるなどツールの整備についても問題が残されていた。

しかし最近では，日立製作所のクリエーティブワークステーション2050に見られるように，高機能，高性能なワークステーションが急速に普及し，

(1)　はん用コンピュータへの機能集中形からワークステーションへの機能分散化

(2)　アイコンやマウスを活用した操作性の向上

(3)　PAD[6)](Problem Analysis Diagram)の作成や変更とソースプログラムの自動生成

などの実現も容易になり，従来の問題点も容易に解決できるようになった。

本稿では，日立製作所のクリエイティブワークステーション2050に搭載したSEWB[7)](Software Engineering Workbench)をベースに，機能分散形マイクロコンピュータ向けソフトウェア開発支援ツールの構成及び適用法について概説する。

2 適用のポイント

ソフトウェアの開発は，システム分析，システム設計，ソフトウェア設計，ソフトウェア製造及びテストの五つの工程に分けられる[8)]。

この中で作業工数が急増する工程は，ソフトウェア設計以降である。現在実用化しているマイクロコンピュータ用開発支援ツールは，ソフトウェア製造工程とテスト工程をツールで支援しているが，ソフトウェア設計工程は，その一部しか支援していないのが実態である。

そこでソフトウェア設計にSEWBを採用し，ソフトウェア製造工程にはPAD化作業以降の自動化をして，設計や製造時

* 日立製作所戸塚工場　** 日立製作所システム開発研究所　*** 日立製作所那珂工場　**** 日立製作所水戸工場

に行う図面の作成を容易にし，かつソースプログラムまでの作業を自動化する環境を実現することにした。

一方，通信システムで採用されている状態遷移図[9]をベースにした状態遷移設計支援ツールは，一般にハードウェアのシーケンスの制御を行うことを主な役目とする一般のマイクロコンピュータ用のソフトウェア設計工程向けに適用できるものである。

2.1　ソフトウェア設計における図形化仕様入力環境の構築

ハードウェアは各種の状態を持ち，この遷移を実現することがソフトウェアの機能となる。したがって，ソフトウェアの要求仕様の表現として状態遷移図が適している。

すなわち，本稿の状態遷移設計支援ツールでは，各種ハードウェアとその制御をそれぞれ順序機械とみなし，各ハードウェアの一つの安定状態で入力を与えることによって，次の安定状態に遷移する状態遷移を図示したものである。

しかし，従来の状態遷移設計支援ツールは，状態遷移図を書き，その後状態遷移仕様記述言語に人手で変換して入力する形態になっており，工数や品質面で問題を残している。この入力での問題解決のために，以下の機能を実現することにした。

(1)　状態遷移図の作成及び変更を容易にするエディタ

(2)　状態遷移図と既存の状態遷移設計支援ツールとの接続

SEWBは基本部と，これにユーザーが機能を追加しやすいツールインタフェースがあって，今回このツールインタフェースを利用し，上記機能を追加した。

2.2　ソフトウェア製造における図形化処理環境の構築

従来は，仕様の決まった部品(モジュール)のプログラム論理をPAD図に書き，その後，図面修正，コーディング及び端末から入力と変換作業を人手によっており，この間の工数や品質の面でロスがあった。

このPAD化作業からコーディングまでを自動化し，前記問題の解決を図るため**表1**に示す機能を実現することにした。

3　適用方法

ソフトウェア設計向けには状態遷移設計支援ツールを適用し，ソフトウェア製造向けにはPAD支援ツールを適用した。

適用に当たり一部機能追加を行い，従来の支援ツールとの整合をとった。詳細を以下に述べる。

3.1　状態遷移設計支援

ソフトウェア設計での作業で大半の工数を占める作業は，外部仕様をシステムの構成する部品に展開する作業である。状態遷移の設計では外部仕様をもとに個々の状態と遷移の関係を明確にし，その結果を状態遷移図として作成する。

この遷移条件や遷移処理の具体化では，最適な関係を見いだすまで試行錯誤を繰り返し，状態遷移図の修正を繰り返す。

次に，状態遷移図を状態遷移設計支援ツールに入力するため，状態遷移仕様記述言語に変換し端末から入力をする。

この作業の生産性向上には状態遷移図の作成や変更が容易にできるとともに，状態遷移図から自動的に遷移ソースプログラムを生成する環境が必要になる。この環境を実現するためSEWBを採用し，状態遷移設計支援ツールを次のとおり改良した。

SEWB導入前の状態遷移設計支援ツール(**図1**)は，状態仕様記述言語(**図2**)で記述したものを入力し，状態遷移ソースプログラム(**図3**)などを生成していた。この状態遷移仕様記述言語入力部分にSEWBを導入し，その図形処理能力を活用して状態遷移エディタ(**図4**)を作成した。その結果，状態遷移図を直接入力できるようになった。

状態遷移エディタの開発に当たっては，操作性を重視して次のような対話インタフェースを採用した。

(1)　一貫したエディタの操作性

ソフトウェア設計者は複数の工程にわたり作業することがあり，設計の対象ごとに使用するエディタが違うのでは効率が悪い。したがって，PADエディタなどSEWB上のツールと同じにした。

(2)　画素(状態遷移図で使用する記号)はCCITT(Consultative Committee of International Telegraph and Telephone：国際電信電話諮問委員会)で勧告されているもの(SDL/GR：Functional Specification and Description Language/Graphic Representation)を採用

(3)　文法誘導形編集方式の採用

状態遷移図は，画素の結合及び配置に一定の規則(文法)を

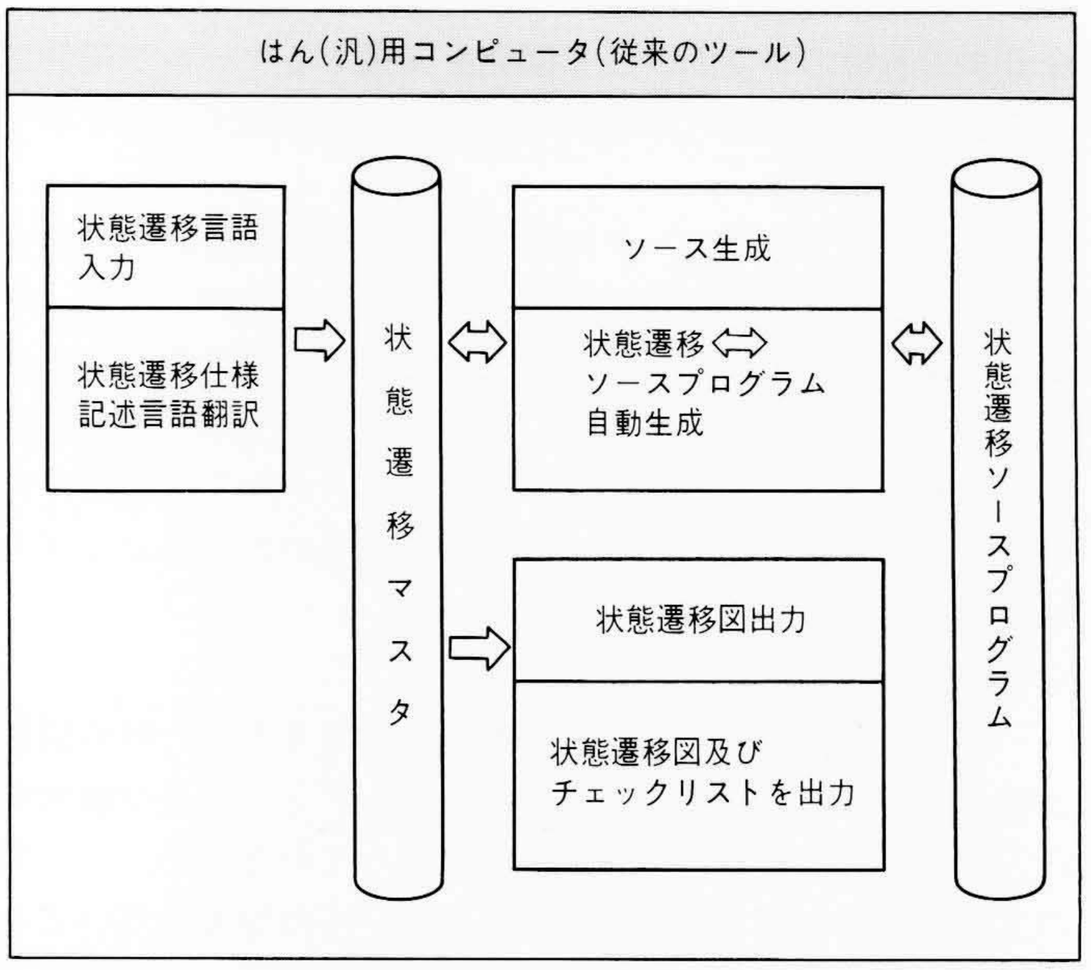

図1　従来の状態遷移設計支援ツールの構成　　図2で示す状態遷移仕様記述言語を入力とし，**図3**で示す状態遷移ソースプログラム及び**図6**で示す状態遷移図を出力する。

表1　ソフトウェア製造工程の作業と実現策　　プログラム論理の図形表現(PAD図)の作成やプログラム言語への変換など，人手介入部で，かつ作業が定形化した部分は自動化の効果も大きい。

主な作業	実現策
部品内部の論理を図形表現(PAD)	●PADの作成及び変更の容易化
	●PADでの論理検証の容易化
プログラム言語に変換	●PADからソースプログラムの自動生成

注：略語説明　PAD(Problem Analysis Diagram)

```
STATE 681 /*xxx*/;
    INPUT CTLKE;
    TASK IDLE;
    OUTPUT COMD(RSTPRCVC);
    OUTPUT MAIL(CMJMP);
    NEXTSTATE 550;
    INPUT CSIGE;
    TASK CONN;
    OUTPUT COMD(RSNDBSGC);
    OUTPUT COMD(RSTPRCVC);
    TASK CIFC(TAC);
    TASK CIFC(CON);
    TASK TRFC(EBAC);
    NEXTSTATE 515;

ENDPROCESS;
```

図2　状態遷移仕様記述言語の例　　CCITT(Consultative Committee of International Telegraph and Telephone)で勧告されているテキスト表現SDL/PR(Functional Specification and Description Language/Textual Phrase Representation)で記述した状態遷移仕様記述言語のコーディング例を示す。

```
        TRTM CTLKE
        TRTM CSIGE
        STOPM

        IDLE
        COMD RSTPRCVC
        MAIL CMJMP
        TEND

        CONN
        COMD RSNDBSGC
        COMD RSTPRCVC
        CIFC TAC
        CIFC CON
        TRFC EBAC
        TEND
```

図3　状態遷移ソースプログラムの出力例　　図2で示した状態遷移仕様記述言語から生成された状態遷移ソースプログラムの出力例を示す。各文はマクロ表現されており，言語プロセッサのマクロ展開機能で展開される。

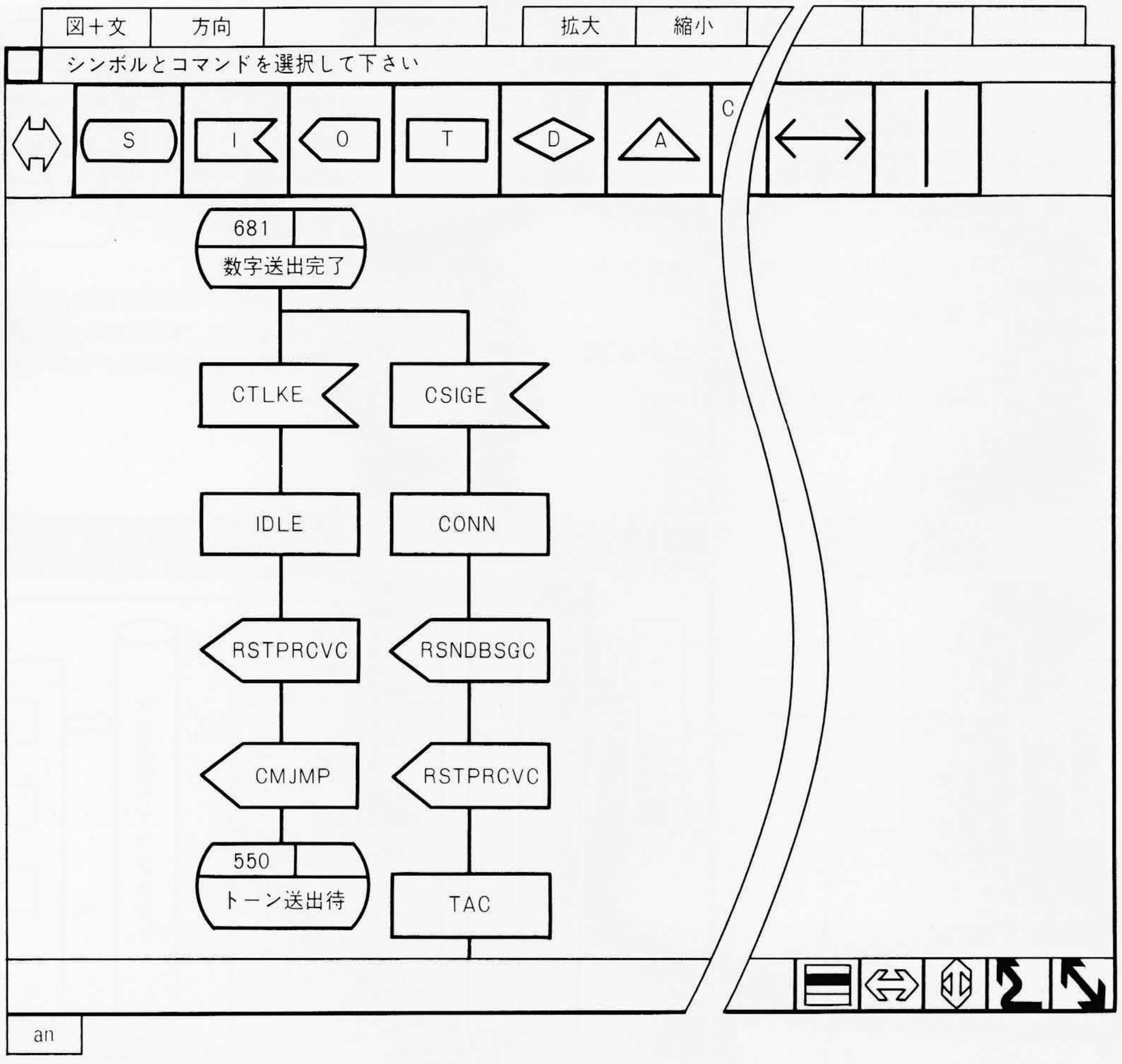

図4　状態遷移エディタの入力例　　図2で示した状態遷移仕様の内容を，SEWB上の状態遷移エディタで直接入力した入力画面の例を示す。各シンボルの図形形状は，CCITTで勧告されているSDL/GR(SDL/Graphic Represention)に準拠している。

持っている。

この規則を状態遷移エディタに内蔵させることによって，結合や配置を自動決定し入力を誘導する方式にした。

以下，状態遷移設計支援ツール(**図5**)を利用したソフトウェア設計作業について詳細を以下に述べる。

(1) 状態遷移図の作成及び変更

SEWB上の状態遷移エディタ(**図4**)を用いて，直接状態遷移図を作成又は変更する。

状態遷移エディタで入力された状態遷移図は，状態遷移マスタに蓄積される。

(2) 状態遷移マスタをはん用コンピュータへ転送

ファイル転送機能を用いて，状態遷移マスタをはん用コンピュータへ転送する。

(3) 状態遷移図の出力

状態遷移図出力機能を用いて，状態遷移図(**図6**)及び各種チェックリストを出力し，論理矛盾などのチェックを行う。

チェックの結果，変更する必要があれば前記(1)に戻る。

(4) 状態遷移ソースプログラムの生成

状態遷移図上の変更作業が終了した後，ソース生成機能を用いてソースプログラム(**図3**)を生成する。

3.2 PAD支援

状態から状態への遷移がプログラムとなる。このプログラムの作成をPAD支援ツールで支援する。

本ツール(**図7**)を利用することで，PADからソースプログラム間の変換作業で発生していたミスは皆無となり，更に，その間の作業工数も大幅に短縮できる。

その理由は，従来人手で書いていたPADをPADエディタで容易に作成できるとともに，このPADからソースプログラムを自動的に生成できることである。

また，机上レビューの作業も，PADテスタで具体的な動作がアニメーションされるため容易になる。

このうち，PADエディタについては，はん用的なものが既にSEWB上に準備してあり新規に作成する必要はない。

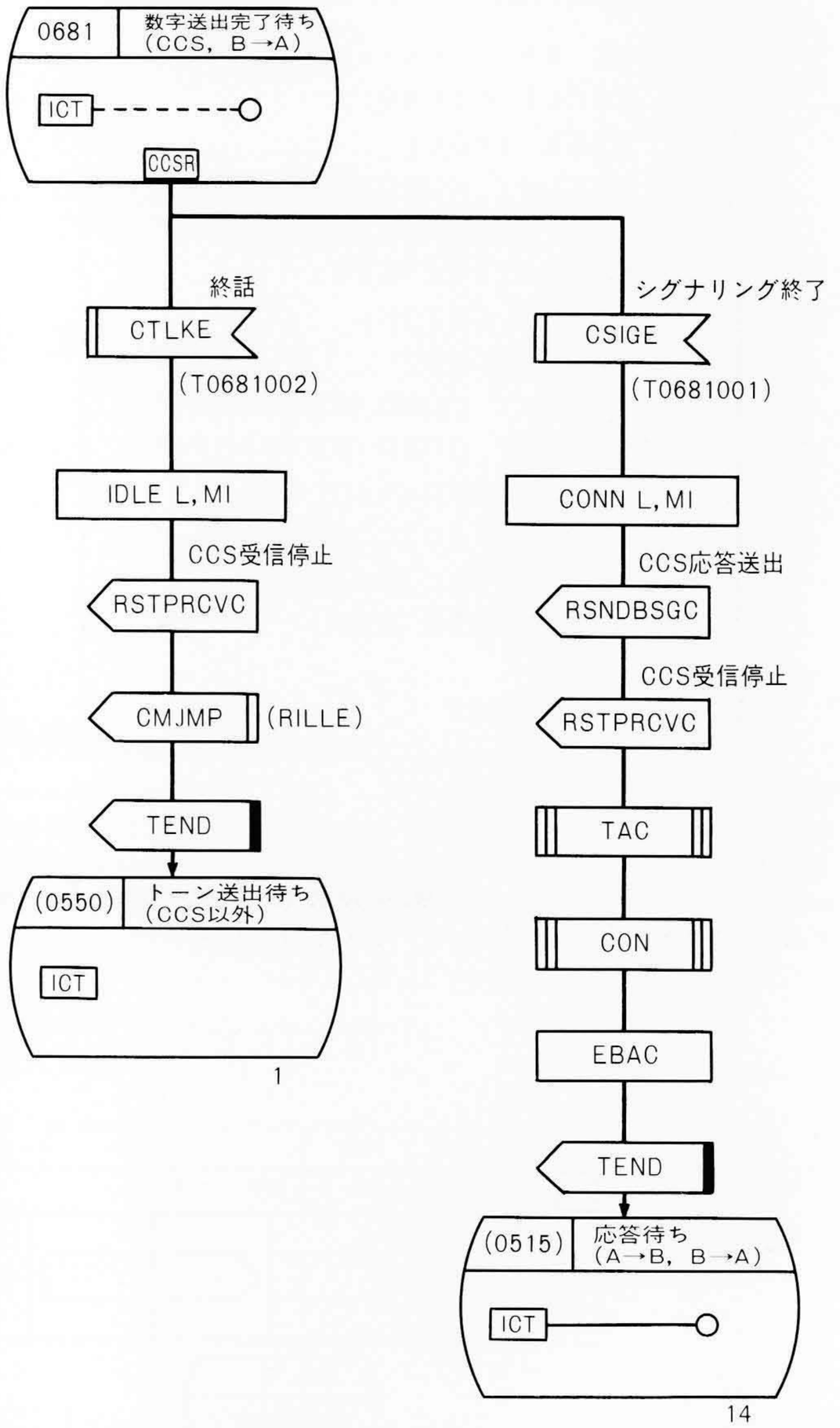

図6 状態遷移図の出力例　状態遷移図出力ツールによって出力された状態遷移図の出力例を示す。本図には，別途図形ライブラリで定義されているコメントや状態内部の詳細図が付加されている。

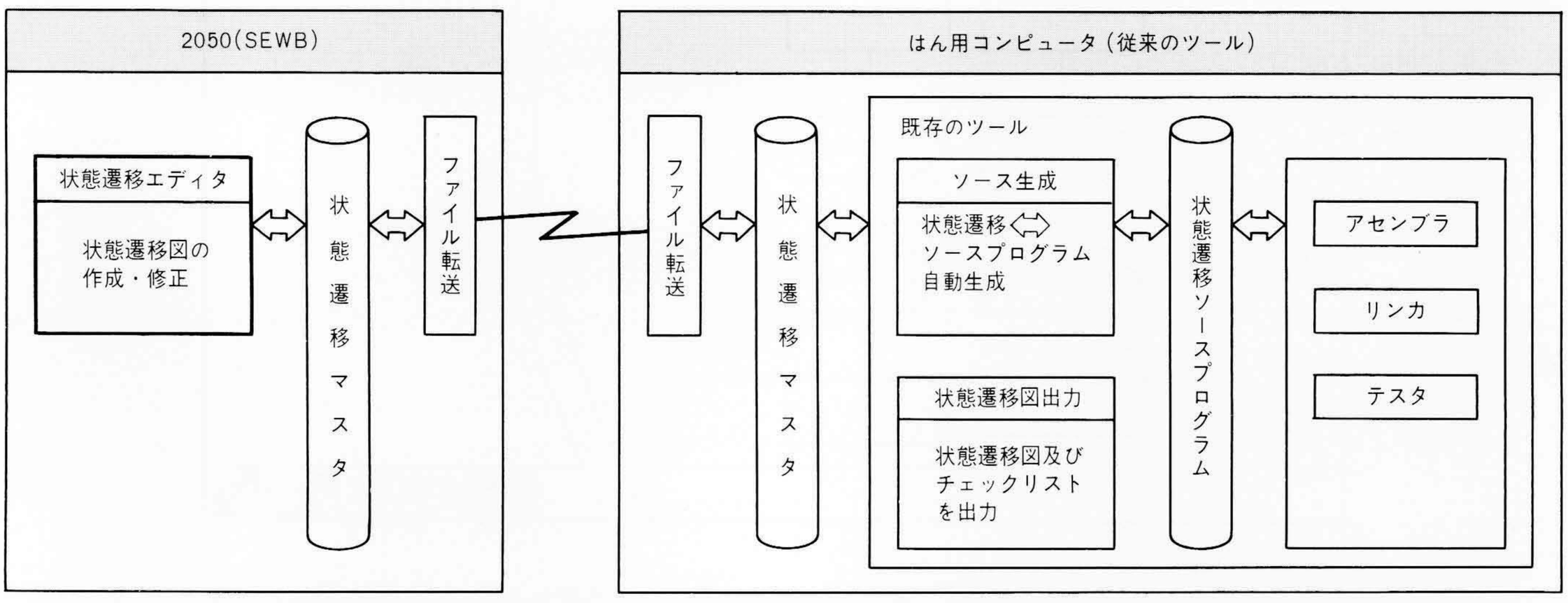

図5 SEWBを利用した状態遷移設計支援ツールの構成　状態遷移エディタをSEWB上に実現し，従来のツールと接続した。SEWBと従来ツールの間は，ファイル転送機能で接続している。

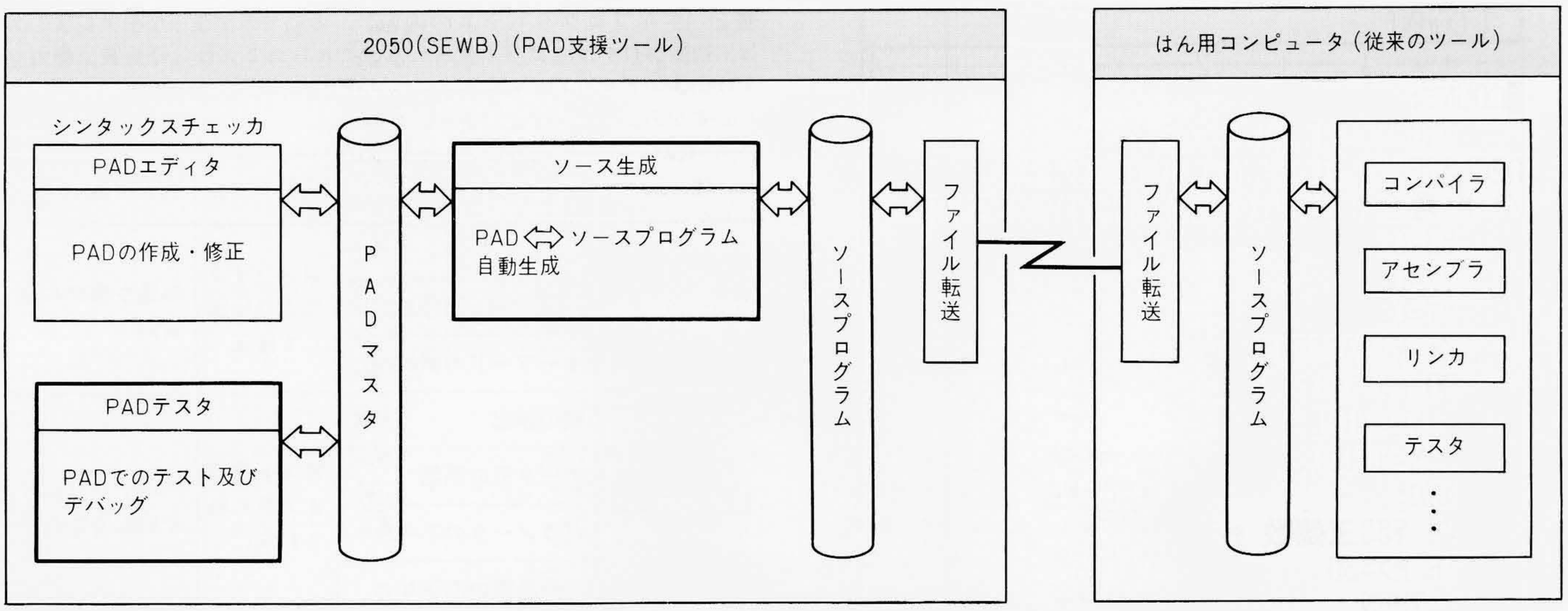

図7　PAD支援ツールの構成　　SEWB上のPADエディタでPADの作成及び修正を行い，論理の正当性をPADテスタで確認する。その後，ソースプログラムを生成する。SEWBと従来ツールの間は，ファイル転送で接続している。

しかし，ソース生成及びPADテスタは，対象としているソースプログラムの言語に依存しているため，マイクロコンピュータ用言語のものを新規に開発し，SEWB上に搭載した。

以下，PAD支援を利用したソフトウェア製造作業は次のとおりである。

(1)　PADの作成及び変更

SEWB上のPADエディタを用いて直接PAD(**図8**)を作成する。PADエディタで入力されたPADは，PADマスタに蓄積される。

(2)　PADレベルでの論理の検証

SEWB上のPADテスタを用いて，PADレベルで実際に論理を動かし，正しさを検証する。もし動作内容が設計意図と異なるときは，その間違い点を動作軌跡を追いながら検出し，前記(1)に戻りPADの修正を行う。

(3)　ソースプログラムの生成

PADテスタによるテストの結果が正しいことを確認したならば，ソース生成機能を用いてソースプログラム(**図9**)を生成する。

4　適用効果

SEWBは図式表現(状態遷移図及びPAD)からソースプログラムを半自動生成することを可能にした。このことにより，初級プログラマにも一定水準のソフトウェア開発を可能にした。

以下に，信頼性向上・生産性向上の効果を挙げた具体的な効果について述べる。

(1)　ソフトウェア品質の向上

上記自動生成により，プログラマの個人差のない均質なソースプログラムを得ることができる。

通常プログラマに多いミスは，文法に起因するものと意味に起因するものがあるが(**表2**)，そのうち文法上のミスはソースプログラムを自動生成するので皆無となる。更に，意味上のミスはPADテスタの使用で検出可能となる。

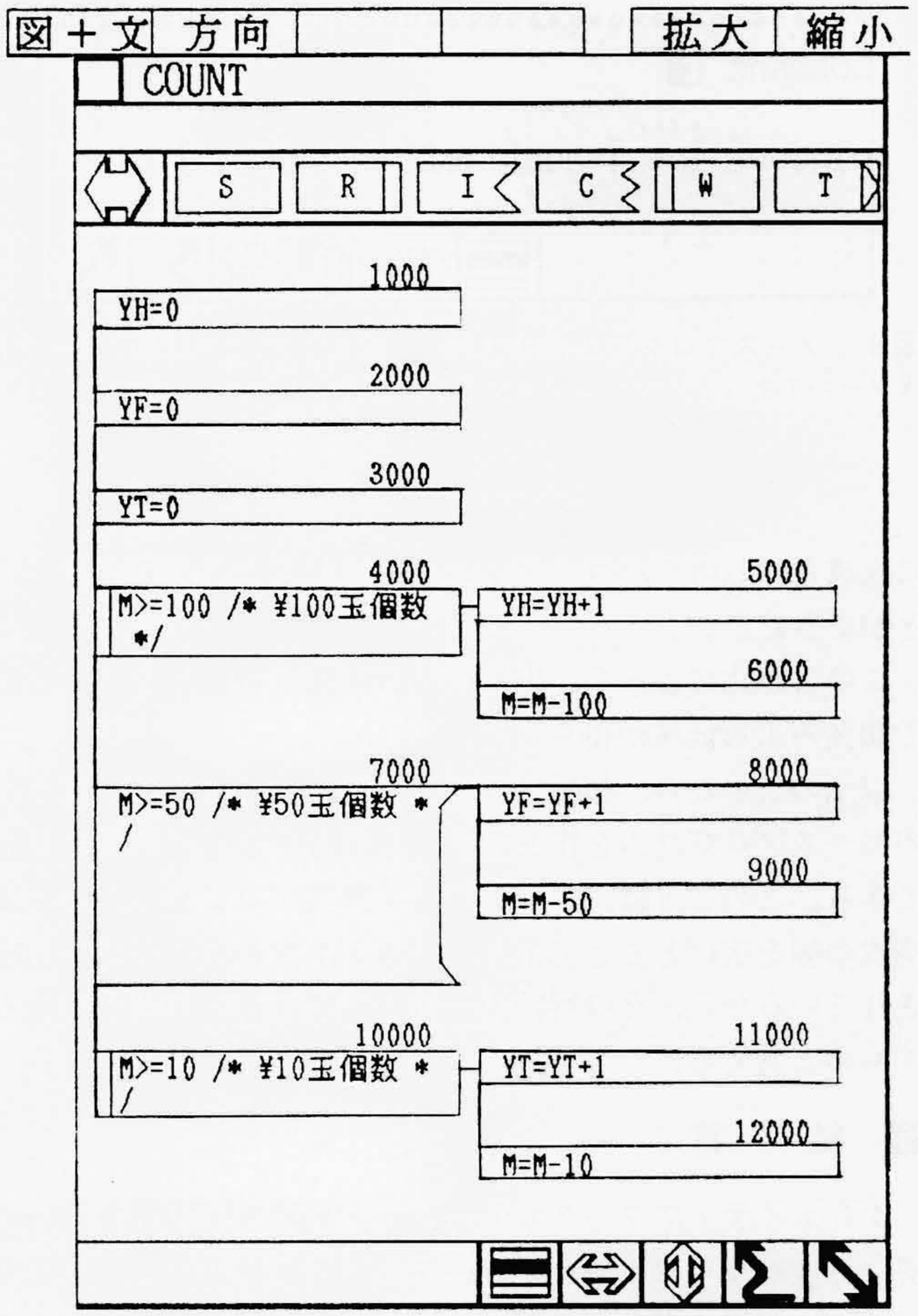

図8　PADエディタによる入力例　　SEWB上のPADエディタを用いて入力したPADの入力例を示す。

(2)　ソフトウェア開発工数の低減

従来のソフトウェア開発では机上でのコーディングと，その端末入力が必要であったが，本ツールでは状態遷移図及びPADを直接入力することによってソースプログラムが自動的

```
COUNT
----+----1----+----2----+----3----+----4
COUNT:
    PROCEDURE (M,YH,YF,YT);
DECLARE (M,YH,YF,YT) INTEGER;
/* M:残金；YH:¥100玉個数；YF:¥50玉個数；
    YH=0;
    YF=0;
    YT=0;
    /* ¥100玉個数 */
    DO M>=100;
      YH=YH+1;
      M=M-100;
    END;
    /* ¥50玉個数 */
    IF M>=50
      THEN
      DO;
        YF=YF+1;
        M=M-50;
      END;
**************************************
Command
window[ 1st ] num[  off ] sep[    ] pipe[
```

図9 ソースプログラムの出力例 図8で示したPADを，ソース生成ツールで生成したソースプログラムの出力例を示す。

に生成されるので，コーディングと端末入力の作業を除くことができた。

この自動化によって，更に期間の短縮も可能となり，全体で30%の工数低減が得られた。

以上のことから，初級プログラマであってもある一定水準のソースプログラムを作成でき，即戦力としての活躍が期待できる。このことは，ソフトウェア開発プロジェクトの人員構成の幅を広げることであり，プログラマの数の不足が懸念されている今日，その解消の一策として大きな効果をもたらすものと言える。

5 結 言

クリエイティブワークステーション2050のSEWBを用いた機能分散形マイクロコンピュータ向けのソフトウェア開発支援ツールの考え方及び適用について述べた。

ソフトウェアの重要性が増え，需要が拡大している今日，生産性向上は極めて重要かつ急を要する問題である。

表2 主なプログラムミスの内容 本表で示すようなケアレスミスは，初級プログラマに多く見られる。これらのミスは，発生源で極力見つけ出すことが必要であり，ツールも高度化，高機能化し，対応していく必要がある。

種 別	プログラムミスの内容	従来ツール	本ツールの適用
文法上のミス	構文の不一致	コンパイラにより検出して修正	自動生成のため皆無
	文の区切り記号の抜け		
	キーワードの誤記など		
意味上のミス	値の誤記	テスト段階まで持ち越される。	PADテスタにより検出できる。
	データ形の誤記		
	パラメータの不一致		
	動作論理の不正など		

上級プログラマの能力を十二分に発揮させることも必要であるが，同時に大多数の中級・初級プログラマを戦力に加えトータルとしての生産力の拡大を図ることが重要である。

図式表現によってソフトウェアを半自動的に生成するSEWBを構築したことは，この要求にこたえるものであり作業のしやすい環境を提供することで，中級・初級プログラマを即戦力とすることを容易にすることができた。

なお，本ツールの開発には，日立ソフトウェアエンジニアリング株式会社及び日立コンピュータエンジニアリング株式会社の協力をいただいた。

参考文献

1) 黒崎，外：通信用ソフトウェア生産技術の高度化，日立評論，**64**，3，171～174(昭57-3)
2) 増井，外：通信用ソフトウェア開発支援システムの一構成法，電子通信学会，SE83-141(昭58-12)
3) 黒崎，外：マイクロコンピュータ用会話形テスト支援システム“HITS”，日立評論，**66**，3，203～206(昭59-3)
4) 田中，外：計算機誘導形構造エディタ，日立評論，**68**，5，373～398(昭61-5)
5) 橋本，外：マイクロコンピュータ用ソフトドキュメント自動生成システム“Auto-DS”，日立評論，**68**，5，361～365(昭61-5)
6) 二村，外：PADの開発，日立評論，**68**，5，351～356(昭61-5)
7) 葉木，外：SEWBの開発思想と機能，日立評論，**70**，2，101～108(昭63-2)
8) 小林，外：ソフトウェア生産技術の最近の動向，日立評論，**68**，5，345～360(昭61-5)
9) 檜山，外：電子交換機の機能分散ソフトウェア方式，日立評論，**68**，5，357～360(昭61-5)