U.D.C. 681.323.06：〔621.395.345：681.325.6〕.001.57

電子交換機論理機能のシミュレーション・システム

Logic Simulation System for Electronic Switching Equipment

菅野文友* 今出英雄**
Fumitomo Kanno Hideo Imade

要　旨

日本電信電話公社 電気通信研究所を中心として，交換機メーカー4社（日本電気株式会社，沖電気工業株式会社，富士通株式会社，株式会社日立製作所）が共同研究中の電子交換機の開発に際し，開発能率向上を目的として，その中央処理装置の方式上および論理設計上の誤りを検出するため，HITAC 5020による，論理機能のシミュレーション・システムを開発した。本システムは高速化と大容量化のため，約2万素子のシミュレーションを5～10秒/クロックで行なった。また15種類の入力条件について，並列処理を行なった。これらに加えて，機能ブロックを論理式によって記述したものを，そのまま翻訳してシミュレーションする，論理機能記述言語も備えている。本システムは電子交換機の開発に多角的に活用され，その有用性が確認された。本稿はシステムの根幹であるシミュレーション方式についてその概要を述べる。

1. 緒　　言

現在，日本電信電話公社 電気通信研究所を中心に，日本電気株式会社，沖電気工業株式会社，富士通株式会社および株式会社日立製作所の交換機メーカー4社の共同研究として，電子交換機の実用化研究を推進中であり，すでに，室内実験用 DEX-1 電子交換機，現場試験用 DEX-2 電子交換機が開発された。

電子交換機の開発に際しては，設計から稼働にいたるまでの開発作業の能率化と開発線表の厳守を目的として，大形電子計算機 (HITAC 5020) による，いくつかの設計自動化がなされた。

論理シミュレーション・システムもその一環として，電子交換機の中枢部分である2～3万個の素子数をもつ，中央処理装置の設計業務の能率化のため開発された。このように大規模な論理装置の開発における，方式設計時および論理設計時のチェックを人手で行なうことは，膨大な工数を要し，また全体にわたる種々の機能のじゅうぶんなチェックを行なうことは不可能に近い。

この論理シミュレーション・システムは方式設計時に機能ブロックを論理式として記述したデータを入力データとしてチェックし，さらに，それらの機能の確認された部分について，論理設計の各素子単位の論理データに置き換えてゆき，金物の製作，調整工程と並行して，論理ミスの摘出と金物のテスト・プログラムの作成を行なうことを目的としている。

なお，本シミュレーション・システムは，LSS-6 (Logic Simulation System-6) と称し，その基本的思想は電話料金処理システム (CAMA) の開発時に活用されたもの[1]をはじめとし，それに数次の改良を加えてきたLSS-4[2]に基づいているが，本システムでは，これらに大幅な機能追加を行なっている。その特長は次のとおりである。

（1） 複数個（15バッチ）の初期条件に対するシミュレーションを並列に行なうことができる。

（2） シミュレーション可能な素子数は約20K程度である。原理的には32Kまで可能である。

（3） シミュレーションの実行制御のための言語(マクロ命令)は63種あり，それにアセンブラ言語を混用することができる。

（4） 論理記述用言語の翻訳機能を持ち，入力データとして素子単位の論理データのほかに，論理式の論理記述用言語で論理構成をマクロに記述することができる。

* 日立製作所ソフトウェア工場　工学博士
** 日立製作所ソフトウェア工場

2. 概　　要

2.1 機能概要

本論理シミュレータへの論理情報の入力方法は二とおりある。

（1） 方式設計時に機能ブロック・レベルで論理仕様を論理記述言語にしたがって，記述した論理構成データ。

（2） 論理設計によって作成される，各論理素子レベルでの接続状態を表わす論理データ。

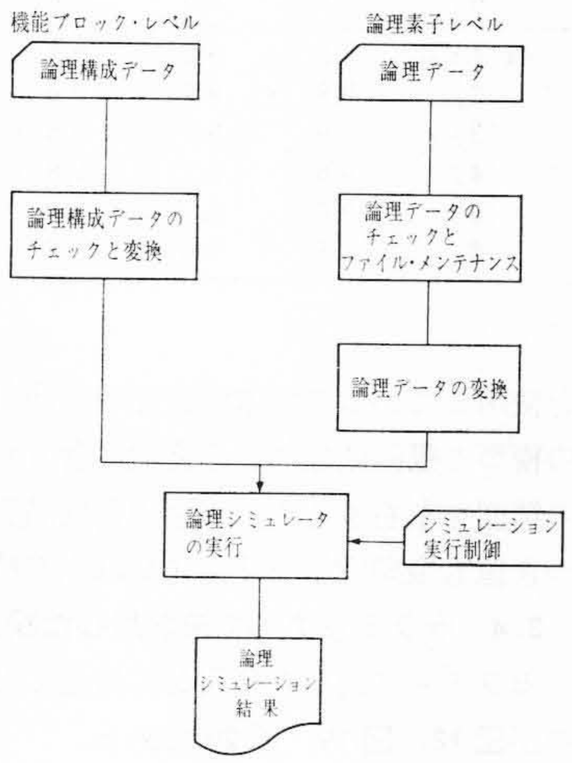


図1　論理シミュレーション手順

これらの入力データによる，論理シミュレーション手順を示したのが図1である。

論理素子レベルのシミュレーションの場合は，

（1） 論理回路情報として，シミュレータに入力するデータのチェック，データの変更，追加，削除および論理データ，ファイルのメンテナンスを行なう。

（2） 論理データをシミュレーションできるデータに変換し，素子の接続状態に合わせて分類する。また，これらのデータを本論理シミュレータではHITAC 5020の機械命令に翻訳する。

（3） 命令形式に翻訳したデータを使用して，シミュレーションを行なう。シミュレーションの実行制御は論理シミュレーション用マクロ命令によって行なわれる。

（4） シミュレーション実行のための初期条件は，シミュレーション用マクロ命令により，任意の時点で任意の素子にセット，リセットしたり，タイム・チャートで与えたり，シミュレータ内の擬似メモリ内にテスト，プログラムとして与えたりすることができる。

（5） シミュレーションの結果はシミュレーション用マクロ命令によって，シミュレーション実行時，または実行後に編集して取り出される。

以上の操作に加えて，機能ブロック・レベルのシミュレーションの場合は

（1） 論理記述言語で記述された，論理構成データを分解し，論理演算，算術演算，シフト演算などの処理ルーチンにリンクするための機械命令および引数に翻訳する。

表1　適用計算機システム(HITAC 5020)の最小機器構成

機器名称	数	備考	
処理装置(65K語)	1	式	HITAC 5020　32 ビット/語
磁気ドラム記憶装置	8	台	65 K語/台　H-179A
磁気テープ装置	8	台	120 K語/台　H-3485
カード読取機	1	台	600 枚/分　H-329
カードせん孔機	1	台	200 枚/分　H-334
ラインプリンタ	1	台	1,000 行/分　H-333

表2　4入力 AND 回路の翻訳プログラム

順序	操作	HITAC-5020 アセンブラ言語による翻訳プログラム		状態の変化状況
1	素子Bの状態の1の補数をとる。	CS	r1, B	11001
2	素子Dの状態の1の補数をとる。	CS	r2, D	11101
3	1と2のビット単位の乗算	MB	r1, r2	11001
4	素子Aの状態と3をビット単位の乗算	MB	r1, A	01001
5	素子Cの状態と4をビット単位の乗算	MB	r1, C	00001
6	5を素子Eの状態表にセット	T	r1, E	00001

注 1.　CS : CLEAR SUBTRACT
　　MB : MULTIPLY BITWISE
　　T : TRANSFER
注 2.　r1, r2 はレジスタ

(2)　前項(2)で作成された，翻訳データと合わせて，前項(3)の処理を行なう。

論理データと論理構成データを混合して，シミュレーションすることもできるし，おのおの単独であってもさしつかえない。

2.2　適用計算機システム

適用計算機システム，HITAC 5020 の最小機器構成は表1に示すとおりである。

3. シミュレーション方式

3.1　並　列　処　理

論理シミュレーションにより，論理ミスが検出されると，論理データに修正を加え，さらにシミュレーションを行なうが，このサイクリックな操作に要する時間，すなわちターン・アラウンド・タイムの大きさが論理シミュレーションの効果を大きく左右する。

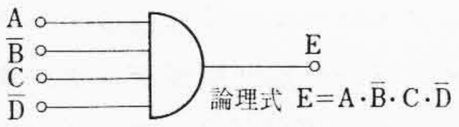
本シミュレータでは，ターン・アラウンド・タイムを小さくするため，同一論理に対し，相異なる15種類の条件についてのシミュレーションを，1種類の条件で行なうと同じ時間で，行なうことができるよう考慮した。

図2に示す AND 回路の例で基本的な方法を説明する。各論理素子の状態表を図2のように1素子に対して，コアメモリ1語 (32 ビット)を割り当て，1語の先頭から15ビットをクロックnの状態表，右半語の先頭から15ビットをクロック (n+1) の状態表とする。フリップ・フロップなどの記憶論理はクロックnの状態表の値を入力とし，出力をクロック (n+1) の状態表に入れ，一般論理は過渡的な値のみを必要とするため，クロックnの状態表の値でシミュレーションを行ない，結果をクロックnの状態表に入れる。このようにして1素子当たり，15種類の状態表を確保している。

図2の4入力 AND 回路をシミュレートするための翻訳プログラムの実行順序およびその状態の変化状況は表2に示すとおりである。表2のように，素子の機能および入力名から，アセンブラ言語に翻訳し，それを実行することにより，15種類の条件に対するシミュレーションを行なうことができる。なお，状態表の最下位を1にしてあるのは1の補数を HITAC 5020 の一命令で実行するためである。

3.2　翻訳プログラム形式

論理シミュレータでは一般に解釈形式と翻訳形式があるが，本シミュレータではシミュレーション速度の観点から翻訳形式を採用している。



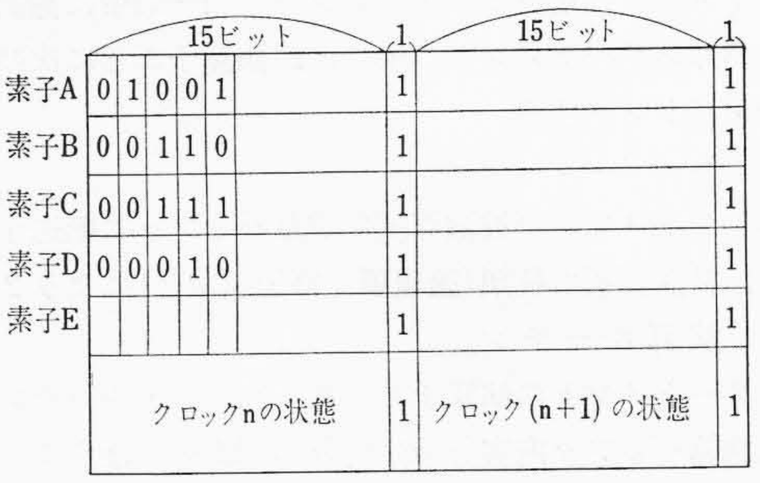
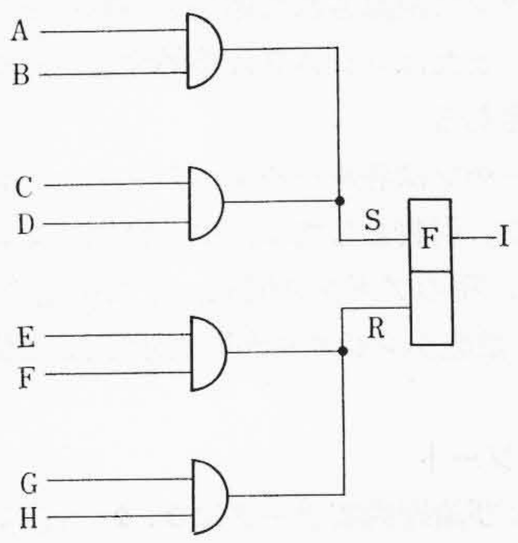


図2　4入力 AND 回路と素子状態表



翻訳形式

セット・リセット形フリップ・フロップ・サブルーチン

```
CA   r1, A
MB   r1, B
CS   r1, r1
CA   r2, C
MB   r2, d
CS   r2, r2
MB   r1, r2
CS   r1, r1
CA   r2, E
MB   r2, F
CS   r2, r2
CA   r3, G
MB   r3, H
CS   r3, r3
MB   r2, r3
CS   r2, r2
JS   FFSR
```

Sの計算

Rの計算

セット・リセット形サブルーチンにリンク

```
CA   r3, r1
CA   r4, r2
MB   r3, r4
JNH  r3, ERROR
CS   r1, r1
CS   r2, r2
CS   r3, F
MB   r3, r1
CS   r3, r3
MB   r2, r3
VCA  r2
     F))16,5
```

S=R=1 をエラーとする

Iの計算

結果の格納

注　CA : CLEAR ADD
　　CS : CLEAR SUBTRACT
　　MB : MULTIPLY BITWISE
　　JS : SUBROUTINE JUMP
　　JNH : JUMP ON NON ZERO
　　VCA : VARIABLE LENGTH CLEAR ADD

図3　セット・リセット形フリップ・フロップの翻訳形式

(1)　素子レベルの論理データの翻訳形式

本論理シミュレータで取り扱うことのできる基本論理回路は次の11種で，同期式論理回路にも適用できる。

(a)　セット・リセット形フリップ・フロップ
(b)　セット・優先形フリップ・フロップ
(c)　AND 論　理
(d)　ŌR　論　理
(e)　NAND 論 理
(f)　NŌR 論　理
(g)　コレクタ共通論理
(h)　外 部 端 子
(i)　ケーブル・ドライバ
　　(クロックのみの入力を含む)
(j)　出力端子
(k)　キー，キー・レシーバ

これらのうち，AND 論理の翻訳形式については，3.1 で述べたが，(c)～(k)もほぼ同様の処理を行なう。フリップ・フロップについては図3に示すように，セット側入力値とリセット側入力値を求め，それらの情報を持って，サブルーチンにリンクし，セット側入力とリセット側入力がともに1となるエラー条件のチェックと，結果の記憶を行なうことにより，共通化による翻訳プログラムの節約を行なっている。

（2）　機能ブロック・レベルの論理構成データの翻訳形式

機能ブロック・レベルの論理構成データの場合，翻訳形式と解釈形式の混合形式であり，GŌ TŌ ステートメントやCŌNTINUEステートメントの場合は，アセンブラ言語—命令に翻訳するが，一般の演算ステートメントの場合は，演算子ごとに次の3種類のルーチンにリンクする。

（a）　ロード・ルーチン

ステートメントの被演算項を演算バッファに転送するルーチンで，被演算項の種類(論理項，算術項など)により2種ある。

（b）　演算ルーチン

ステートメントの演算子により，ロード・ルーチンで転送された演算バッファ内容と演算項の演算を行なうルーチンであり，演算子に対応して，17種ある。

（c）　ストア・ルーチン

演算ルーチンで演算された結果を演算バッファから，中間結果格納場所，またはレジスタに格納するルーチンで，演算項に対応して2種ある。

これらのルーチンは各ルーチンへのリンケージ命令と引数により構成している。引数としては，項の種別（レジスタ名，中間変数名，素子名)，項のアドレス情報，けた指定の方法(下位と上位のけた指定が，数値かレジスタかの表示子)，上位と下位のけた数などである。

3.3　レベル・ソート

論理データおよび論理構成データは**3．2**で述べたように翻訳するが，これに先だって，翻訳プログラムの実行を論理回路の接続状態に合わせておく必要がある。

機能ブロック・レベルの論理構成データの場合は，論理記述言語で記述した順序で実行するので問題ないが，素子レベルの論理データの場合，各データは素子ごとに個別のデータであるため，素子の接続状態に合わせて配列し直さなければならない。

配列方法はフリップ・フロップ，外部端子，ケーブル・トライバ

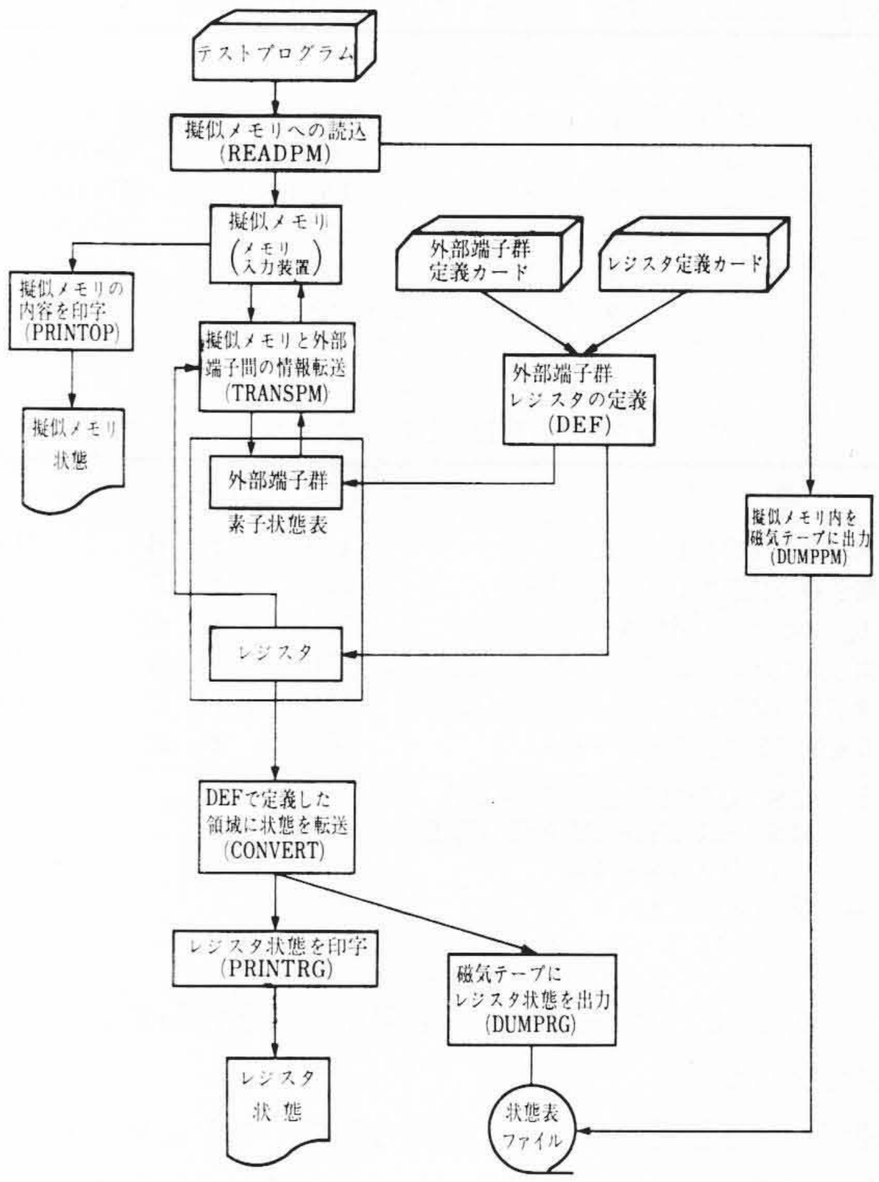


図4　擬似メモリの処理

(クロックのみの入力)の記憶機能を有するか，またはそれに準ずる機能を持つ素子を原点とし，レベル0とする。さらにレベル0の素子を入力とする素子を全論理データから抽出し，レベル1とする。次にレベル0，1の素子を入力とする素子を抽出し，レベル2とする。このような方法で全論理データをレベル付けし，抽出データがなくなるまで，この操作を繰り返す。その結果，抽出されずに残った素子があると，その素子はほかの素子との接続状態が定義されていな

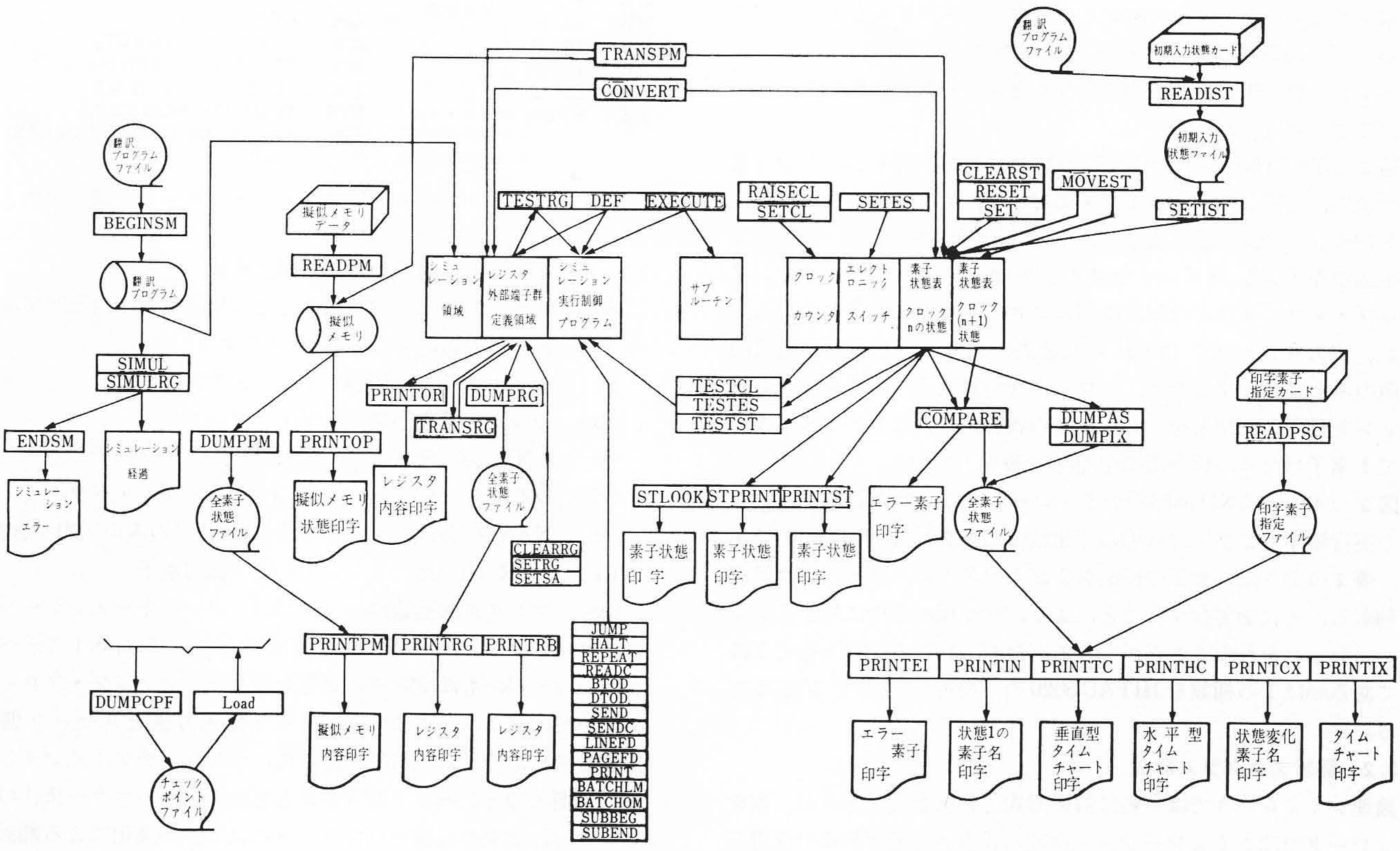


図5　シミュレーション実行制御マクロ命令関連図

表3　マクロ命令機能一覧

分類	機能分類	マクロ命令	機能
オンライン シミュレーション用マクロ (32種)	制御	RAISECL	クロック・カウンタの値を1ずつ増す。
		SETCL	クロック・カウンタに値をセットする。
		SETES	指定エレクトロニック・スイッチのセット，リセットを行なう。
		TESTCL	クロック・カウンタの値を調べ，条件によって実行順序を変える。
		TESTES	指定エレクトロニック・スイッチの状態を調べ条件によって実行順序を変える。
		TESTRG	指定レジスタの状態を調べ，条件によって実行順序を変える。
		TESTST	指定素子状態表を調べ，条件によって実行順序を変える。
	初期設定	CLEARRG	指定レジスタをクリアする。
		CLEARST	素子状態表をクリアする。
		RESET	指定素子の状態表を0にする。
		SET	指定素子の状態表を1にする。
		SETIST	磁気テープ内の「READIST」で作成した初期条件を状態表にセットする。
		SETRG	指定レジスタに指定情報をセットする。
		SETSA	読み込まれた擬似メモリ・データのスタート・アドレスを指定する。
	データ読込	READPM	擬似メモリ内にデータを読み込む。
	シミュレーションの実行	SIMUL	素子レベルの論理データの翻訳プログラムを実行する。
		SIMULRG	機能レベルの論理構成データの翻訳プログラムを実行する。
	転送	CONVERT	レジスタ，外部端子群と素子状態表の間で情報転送を行なう。
		MOVEST	素子状態表のクロックnと(n+1)の間で情報転送を行なう。
		TRANSPM	レジスタ外部端子群と擬似メモリ間で指定レジスタ情報による起動で情報交換を行なう。
		TRANSRG	指定レジスタ間で情報転送を行なう。
	印刷	PRINTOP	擬似メモリ内容を印字
		PRINTOR	指定レジスタ内容を印字
		PRINTST	指定素子内容をを印字
		STLOOK	指定素子内容をタイムチャート形式で印字
		STPRINT	素子状態表の指定範囲を印字
	磁気テープ出力	DUMPAS	素子状態表を出力ファイルに出力する。
		DUMPIX	印字素子ファイルに指定された素子について出力ファイルに出力する。
		DUMPPM	擬似メモリの内容を出力ファイルに出力する。
		DUMPRG	指定レジスタ内容を出力ファイルに出力する。
	その他	COMPARE	素子状態表のクロックnと(n+1)の指定範囲を比較し，異なる素子を出力し実行順序を変える。
		DEF	レジスタ，外部端子群を素子群と対応して定義する。
オフラインシミュレーション用マクロ (11種)	データ作成	READIST	初期入力条件データを編集し，初期入力ファイルを作成する。
		READPSC	印字素子指定データを編集し，印字指定ファイルを作成する。
	印刷	PRINTCX	出力ファイルから状態変化素子の印字をする。
		PRINTEI	出力ファイルからCOMPAREによるエラー素子名を印字する。
		PRINTHC	出力ファイルから素子状態の水平型タイムチャートを印字する。
		PRINTIN	出力ファイルから素子状態1の素子名を印字する。
		PRINTIX	DUMPIXで出力された内容素子状態をタイムチャート形式に印字する。
		PRINTPM	出力ファイルから擬似メモリの内容を印字する。
		PRINTRB	DUMPRGで出力されているレジスタ内容を印字する。
		PRINTRG	出力ファイルからレジスタ内容を印字する。
		PRINTTC	出力ファイルから素子状態を垂直形タイムチャートを印字する。
共用マクロ (20種)	制御	BEGINSM	シミュレーション実行開始処理をする。
		ENDSM	シミュレーション終了処理をする。
		EXECUTE	アセンブラ言語のプログラムとのリンケージを取る。
		HALT	一時停止する。
		JUMP	指定アドレスにジャンプする。
		REPEAT	指定ルートについて指定回数繰り返す。
	データ読込	READC	カード読み込み領域に1枚のカードを読み込む。
	転送	BTOD	2進データを10進に変換する。
		DTOB	10進データを2進に変換する。
		SEND	字単位にデータの転送をする。
		SENDC	指定文字を転送する。
	印刷	LINEFD	指定行だけ改行する。
		PAGEFD	指定ページだけ改ページする。
		PRINT	指定領域情報の印字する。
	磁気テープ出力	DUMPCPF	チェック・ポイント，ダンプを取る。
	その他	BATCHLM	指定ラベルのマクロ命令のバッチ名を指定順序で変更する。
		BATCHOM	次のマクロ命令のバッチ名を変更する。
		PRINTBF	プリント領域に任意文字をセットする。
		SUBBEG	サブルーチン開始マクロ
		SUBEND	サブルーチン終了マクロ

いものであり，誤りである。また，素子の遅延時間の許容レベル内にはいらないものがあれば，これも論理ミスとして検出することができる。

このレベル・ソートでは全論理データの入力名で抽出された素子名を探索するので，テーブル・ルックアップ方法では，膨大な処理時間を要する。そのため，本論理シミュレータでは，素子名を乱数化し，それをアドレスとして，32 K 語のテーブルに格納する方法を取ることにより，探索回数を少なくする。この方法により，約2万素子の場合の処理時間は約7分間である。

3.4 擬似メモリ

初期入力条件の状態表へのセットには，シミュレーション実行時にマクロ SET, RESET により，直接セットする方法とタイム・チャート形式などの初期入力状態データをマクロ READIST によって，磁気テープにいったん作成しておき，シミュレーション時にマクロ SETIST によってセットする方法もあるが，並列処理の場合，能率が悪いことと，テスト用プログラムから，初期入力状態表データに変換する作業量が非常に大きいことにより問題がある。そのため，擬似的にメモリを磁気ドラム上に設け，命令形式のデータをマクロ READPM によって読み込み，これを初期入力条件でセットする方法を取っている。擬似メモリの処理方法は図4に示すとおりである。

(1) 擬似メモリの種類および容量は次のとおりである。電子交換機のメモリ1語を HITAC 5020 の1語に対応させ，ドラム内に各15バッチ分確保する。

メモリ　　4,096 語×15 バッチ

入力装置用　16 語×16 スタック×15 バッチ

入力装置用は論理装置に接続される入力装置の擬似に使用される。

(2) 外部端子群およびレジスタ類はマクロ DEF で状態表内の各素子と対応して定義される。

(3) 擬似メモリへのテストプログラム（擬似メモリ用データ）の読み込みは，マクロ READPM で行なわれる。

(4) 外部端子群と擬似メモリとの間の情報転送はマクロ TRANSPM で行なわれ，その転送は素子状態表内の指定された同期信号によって起動し，その信号の生じた次のクロックで，指定したレジスタによって擬似メモリのアドレスが指定されるが，情報の転送は指定遅れ時間のあとで行なわれる。

(5) 素子状態表内のレジスタの状態の印字はマクロ CONVERT によって，マクロ DEF によって定義されている領域に編集しておき，これをシミュレーション実行時にマクロ PRINTŌR

で行なうか，またはマクロDUMPRGにより，磁気テープに出力しておき，あとでマクロPRINTRGにより，バッチ単位に編集して印字する。

（6） 擬似メモリ内の状態の印字も，シミュレーション時にマクロPRINTŌPによって行ない，あとでバッチ単位に編集して印字する場合，マクロDUMPPM で磁気テープに出力し，マクロPRINTPM で出力する。

3.5 シミュレーションの実行制御

シミュレーションの実行制御は63種のマクロ命令で行なわれる。マクロ命令は大別して，シミュレーションの前処理，後処理のためのオフライン・シミュレーション用マクロ，シミュレーション実行処理のための，オンライン・シミュレーション用マクロおよびオンライン，オフライン共用マクロに分けられる。各マクロの機能を**表3**に示す。また各マクロの関連を**図5**に示す。

3.6 機能ブロックの論理記述言語

（1） 論理構成の記述のための，固有名としては，素子名，レジスタ名，サブレジスタ名，関数名，フォーマル・パラメータ名およびラベル名を使用することができる。

（2） レジスタは一般にけた指定をして使用される。

ABC (N1/N2)

ここで，ABCはレジスタ名，N1とN2はけた指定なしのレジスタ，または10進の整数である。

（3） 定数には，算術定数と論理定数がある。

（4） 算術式の算術項は素子，レジスタ，関数あるいは算術定数のいずれかである。また，算術演算子としては，加減乗がある。

（5） 論理式の論理項は素子，レジスタ，関数あるいは論理定数のいずれかである。また論理演算子としては，否定，論理積，論理和，Module 2 Sumがある。

（6） けた移動演算子は，一般けた移動およびサーキュラについて，左右の計4種がある。

（7） 比較式の比較項は，素子，定数，あるいは，けた指定なしレジスタのいずれかで比較演算子は6種ある。

（8） 演算式の右辺には，条件付きとそうでないものがある。条件付演算式は次の例のとおりで，比較式が成り立つとき，演算式を実行する。

＝演算式　IF（比較式）
＝演算式　IF（比較式）
⋮
＝演算式　IF（比較式）

（9） 演算式の左辺には遅延して実行する表示がある。記述例は次のとおり

AAA 1 (AAA 2/8) ; 10＝演算式

演算式を実行後，10クロック以後，初めて実行されるとき，レジスタAAA 1のAAA 2で示すけた位置からけた位置8に結果が移される。

(10) ステートメントには，GŌ TŌ ステートメント，IFステートメント，CŌNTINUE ステートメントおよび前述の演算ステートメントがある。

(11) 宣言には，レジスタ，サブレジスタおよび関数の宣言がある（**表4**参照）。

(12) いくつかのステートメントをまとめて，ステートメントグループとして定義し，マクロ SIMULRG で実行する（**表4**参照）。

表4　論理構成の記述例

```
C 1 | ラベル欄 11 | 12 | ステートメント欄 13 ... 72 | ID 73 75 | 通し番号 76 80
    | RAAA       |    | TRANSLATE                  |          |
    |            |    | REGISTER                   |          |
    |            |    | AAAA, ....., UAAAA(5), ... |          |
    |            |    | ....., ZBBBB(CC000, ....., EE004) |   |
    |            |    | END                        |          |
    | RKKK       |    | REGISTER                   |          |
    |            |    | KKKKK, ....., VDDDD(10),   |          |
    |            |    | ....., ZDDDD(EE000, ....., EE009) |   |
    |            |    | END                        |          |
    |            |    | SUBREGISTER(014), BCDEF=VBBBB(619), ... |   |
    |            |    | ....., GHIJK=XCCCC(10/11)  |          |
    |            |    | END                        |          |
    |            |    | FUNCTION                   |          |
    |            |    | NAND2(XY, YX)=$(XY*YX)     |          |
    |            |    | NAND3(XY, YZ, ZX)=$(XY*YZ*ZX) |       |
    |            |    | SUM(XY, YX)=XY, A, YX      |          |
    |            |    | END                        |          |
    | XXYY       |    | STATEMENT                  |          |
    |            |    | AAAAA=ⓐAB000*BBBBB         |          |
    |            |    | UEEEE=(UAAAA+UBBBB)*UDDDD  |          |
    |            |    | END                        |          |
    | YYXX       |    | STATEMENT                  |          |
    |            |    | VDDDD=VAAAA#VBBBB          |          |
    |            |    | PPPPP=FFFFF*SSSSS          |          |
    |            |    | END                        |          |
    |            |    | END                        |          |
```

（←1レジスタグループ→ ←レジスタ名宣言→ ←サブレジスタ名宣言→ ←関数宣言→ ←1ステートメントグループ→ ←仕様記述→ ←論理構成記述→ ←機能記述→）

4. システム構成と処理の流れ

4.1 構　　成

論理シミュレーション・システムの構成およびその機能を**表5**に示す。全体のステップ数は約50,000ステップである。

4.2 シミュレーション処理

シミュレーションを行なう場合の処理手順を**図6**に示す。論理装置の方式設計時，または終わった時点で，機能ブロックを論理記述言語で記述した論理構成データを作成する（**図7**参照）。これらのデータは，TRANSLATIŌNジョブによって，レジスタ名称やけた指定などについての，論理構成上の誤りや文法上の誤りのチェックを行ない，アセンブラ言語の翻訳プログラム・ファイルを作成する。一方，方式設計上の問題点について，どのようなシミュレーションを行なうかを考慮し，シミュレーション実行制御プログラムをシミュレーション用マクロ，または必要な場合には，アセンブラ言語を混用して作成する。**図8**はフローチャートの一例を示したものである。これらは，ASSEMBLEジョブによって，アセンブラ言語におとされる（**図9**参照）。実行に先だって，テスト・プログラムを作成し，擬似メモリ・データとする。シミュレーション実行制御プログラム・ファイルおよび翻訳プログラム・ファイルによってシミュレーションを行なう。**図10**はシミュレーション実行後，マクロPRINTRBによって，各レジスタ内容を印字した結果である。これらの結果によって，方式設計段階の誤りを摘出し，論理設計に移

表5　論理シミュレーション・システムの構成

LSS-6-	FILE MAINTENANCE ジョブ	TRANSFER	フェーズ	(論理カードを入力として，論理ファイルを作成する。)
		CHECK	フェーズ	("TRANSFER" でチェックできなかった論理カードの内容のチェックを行なう。)
		LIST 1	フェーズ	(論理素子リストをプリントする。)
		LIST 2	フェーズ	(FAN OUT リストをプリントする。)
		LIST 3	フェーズ	(論理素子辞書をプリントする。)
		EXTRACT	フェーズ	(論理ファイルより，指定グループ名の論理データを抽出し，新論理ファイルを作成する。)
		CORECT	フェーズ	(論理ファイルに対し，データの挿(そう)入，削除，変更を行なう。)
	DATA ARRANGEMENT ジョブ	ILLEGAL	フェーズ	(論理ファイルの各素子の入力名を調べ未定義素子があれば外部端子カードとして，パンチアウトする。)
		SORT	フェーズ	(ILLEGAL フェーズで作成されたデータを分類する。)
		COMPILE	フェーズ	(SORT フェーズで作成された分類データから，命令形式の翻訳プログラムを作成し，LSS COMPILED ファイルを作成する。)
	TRANSLATION ジョブ			(論理構成データカードから翻訳プログラムを作成する。)
	ASSEMBLE ジョブ			(シミュレーション実行制御言語のプログラムカードをアセンブラ言語に変換して，オブジェクト・プログラムファイルを作成する。)
	SIMULATION ジョブ	OFF LINE SIMULATION		前処理（初期状態カードおよび印字素子名カードから初期条件ファイル，印字素子ファイルを作成する。)
		ON LINE SIMULATION		(オブジェクト・プログラムファイルに従ってシミュレーションを行ない，マクロ "DUMPAS"，"DUMPRG"，"DUMPPM" により全素子状態表ファイルを作成する。)
		OFF LINE SIMULATION		後処理（オブジェクト・プログラムファイルに従い，全素子状態表ファイル印字素子ファイルにより，シミュレーション結果をプリントする。)

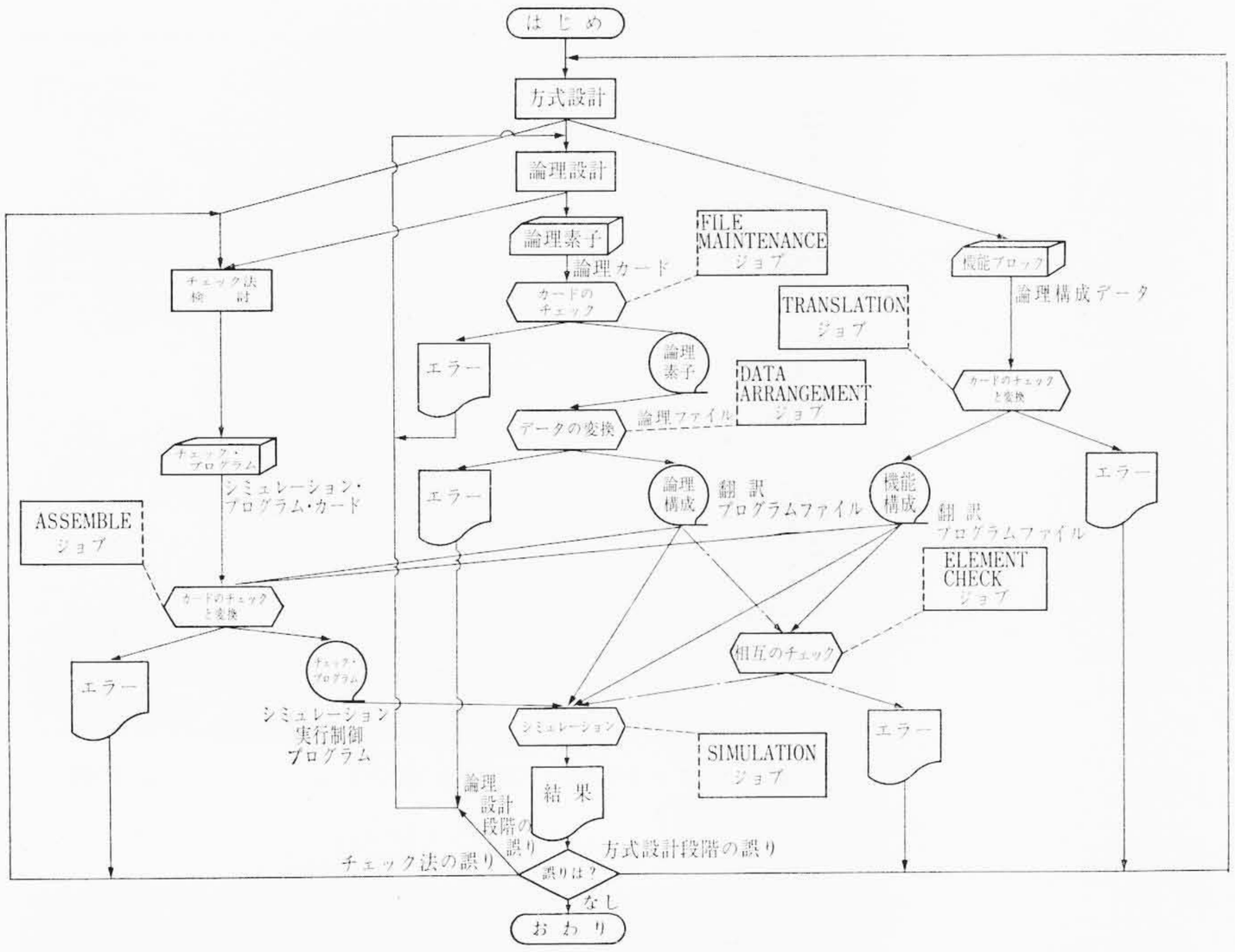


図6　論理シミュレーション手順

行する。

本シミュレータでは，機能ブロック・レベルと素子レベルのデータを混用して，シミュレーションができるので，機能ブロックのうち，部分的に論理設計が完了すれば，それらの論理データで，シミュレーションを行なうことができる。

論理設計が完了すれば，すべてを論理データとする。論理データは，FILE MAINTENANCE ジョブの TRANSFER フェーズで，各素子単位のフォーマット・チェックを行ないながら，論理データ・ファイルを作成し，これを DATA ARRANGEMENT ジョブの ILLEGAL フェーズで，各論理素子の入力に未定義素子があるか，どうかをチェックする。次に，SŌRT フェーズで，素子の段数および素子間の接続状態をチェックし，レベル・ソートを行なう。機能ブロック・レベルの論理構成データと合わせてシミュレーションを行なう場合は，この出力と論理構成データファイル間で相互参照を行ない，閉ループの存在を ELEMENT CHECK ジョブでチェックする。レベル・ソート結果は CŌMPILE フェーズで翻訳プログラム化する。シミュレーション実行に擬似メモリデータ以外の初期入力状態データを必要とする場合は，オフライン・シミュレーションでマクロ READI ST によって，初期入力状態ファイルを作成し，シミュレーション実行時の入力とする。図 11 は初期入力状態データである。また，シミュレーション結果は表 3 に示すように，種々の印字フォーマットで出力するマクロがある。

シミュレーション処理速度は，データによって異なるが，電子交換機 DEX-1 の場合，論理素子は約 2 万素子で，1 クロック，5～10 秒（印字量で異なる）であった。また，論理データに一部修正を加え，シミュレーションを行なうに至るまでの時間，すなわち，CŌRRECT フェーズと DATA ARRANGEMENT ジョブに要する時間は約 1 時間であった。

5. 結　　言

日本電信電話公社電気通信研究所を中心に，交換機メーカー 4 社を加えて，推進中の電子交換機の共同開発研究の一環として，論理装置の論理シミュレーション・システムを開発した。本システムは DEX-1 電子交換機の開発で有効に活用された。さらに，DEX-2 電子交換機の開発に先だって，一部の改良および機能ブロック，レベルのシミュレーションの機能を追加し，DEX-2 の論理シミュレーションに使用し，効果をあげた。

なお，本論理シミュレーション・システムでチェックされた論理データは，そのまま，割付チェック・システムの入力とされ，実装設計の割付チェックに利用された。

今後，ますます複雑化する論理装置の設計，自動化において，その基幹となる，論理シミュレーション・システムを並列処理方式の導入による高速化，シミュレーション言語とアセンブラ言語の混用に

```
          IF(DSL=19) ARITHO                                    TS:06800
           CSL1(DSL0/19)=F                                     TS:06810
C         ADDER                                                TS:06900
ARITHO    0 ADD=ERR IF(DADM)                                   TS:06910
            = RP IF( )                                         TS:06920
          0 ADD=RU.A,ADD,A.1 IF(DADM)                          TS:06930
          1   =RU.A,ADD    IF(DADD                             TS:06940
          2   =RU           IF(DOPE=*(DST+DAND+DOR))            TS:06950
          3   =RP           IF(DOPE=*(DSET+DLD))                TS:06960
              =F            IF( )                               TS:06970
           MAB=RA IF(COU+DST*(C40+C41+C42))                     TS:06980
              =SCC IF( )                                        TS:06990
          END                                                  TS:07000
FF1       STATEMENT                                            TS:08000
C         MEMORY WINDOW                                        TS:08010
           ROM=MWB IF(COU=C13)                                 TS:08020
           RI =MWB IF(*COU=C13)                                TS:08030
           RU =ADD IF(STRW=C41)                                TS:08035
           IF(*(DEST=C41)) NODEST                              TS:08040
C         DESTINATION REGISTER SET                             TS:08100
           RU=ADD IF(IM=0)                                     TS:08110
           R1=ADD IF(IM=1)                                     TS:08120
           R2=ADD IF(IM=2)                                     TS:08130
           R3=ADD IF(IM=3)                                     TS:08140
C         INDEX MODIFY                                         TS:08200
NODEST    RX=ADD  IF(C21)                                      TS:08210
          IF(*=31) NOSCC                                       TS:08220
C         SCC MAINTAIN                                         TS:08300
          0 SCC=RX      IF(JUMPOK)                             TS:08310
              =SCC,A,1 IF(.)                                   TS:08320
C         CONDITION FF SET                                     TS:08400
NOSCC     IF(*(DOPER=DCOM=C31)) NOCOMP                         TS:08410
          0 END=F                                              TS:08420
```

図7　論理記述言語による論理構成データ例

```
          DATE 06.09.68      LSS  ASSEMBLED LIST                              PAGE    1

          SOURCE PROGRAM      (= IS OBJECT PROGNAM)

NO.  ADDRESS  OPERATION  OPERAND                                                ID  SEQ.     ERROR MESSAGE

 1   LBS      ASSEM      TESTLBS1,,ON                                           0L101000
     * 0000Q     SEG                                                          0L100010
     * TESTP     CDATA   (10)/TESTLBS1                                        0L100020
     * TESTP     ENTRY   TESTP                                                0L100030
     * BLOCKN    CDATA   (3)/LBS                                              0L100040
     * BLOCKN    ENTRY   BLOCKN                                               0L100050
     *C ***** REGISTER LIST OF TRANSLATOR WAS LOADED AT ASSEMBLE JOB *****    0L100060
     * ONOFF     VDATA   (16)I/0                                              0L100070
     * ONOFF     ENTRY   ONOFF                                                0L100080
     * DATE      CDATA   (16)/ DATE  06.09.68                                 0L100090
     * DATE      ENTRY   DATE                                                 0L100100

 2   BEGIN     BEGINSM                                                        0L101010
     * BEGIN     JS      ,BEGINSM+2                                           0L100110

 3             READPM    BH1,1                                                0L101020
     *           JS      ,READPM+2                                            0L100120
     *           VDATA   (16)I/00001,00001                                    0L100130

 4             PRINTOP   BH1,H,CS,HD,HFF                                      0L101030
     *           JS      ,PRINTOP+2                                           0L100140
     *           VDATA   (16)I/00001,00001                                    0L100150
     *           VDATA   (16)I/00000,00255                                    0L100160
     *           VDATA   (16)I/00001                                          0L100170

 5             CLEARRG   BH1                                                  0L101040
     *           JS      ,CLEARRG+2                                           0L100180
     *           VDATA   (16)I/00001                                          0L100190
     *           VDATA   (16)I/00000,00030                                    0L100200

 6             SETRG     BH1,C43,1                                            0L101050
     *           JS      ,SETRG+2                                             0L100210
     *           VDATA   (16)I/00001                                          0L100220
     *           VDATA   (16)N/C43,I/00001                                    0L100230
     *           VDATA   (32)IB/10000000000000000000000000000000             0L100240
     *           VDATA   (32)IB/00010000000000000000000000000000             0L100250
     *           VDATA   (32)IB/00000000000000000000000000000000             0L100260
     *           VDATA   (32)IB/00000000000000000000000000000000             0L100270

 7             SETCL     BH1,0                                                0L101060
     *           JS      ,SETCL+2                                             0L100280
     *           VDATA   (16)I/00001,00000                                    0L100290

 8   LOOP      SIMULRG   BH1,FF1                                              0L101070
     * LOOP      JSV     ,MONITOR1                                            0L100300
     *           CDATA   (10)/LOOP                                            0L100310
     *           JS      ,SIMULRG+2                                           0L100320
     *           VDATA   (16)I/00001                                          0L100330
     *           JSS     ,FF1                                                 0L100340

 9             SIMULRG   BH1,LOGIC                                            0L101080
     *           JS      ,SIMULRG+2                                           0L100350
     *           VDATA   (16)I/00001                                          0L100360
```

図9　シミュレーション実行制御プログラム例

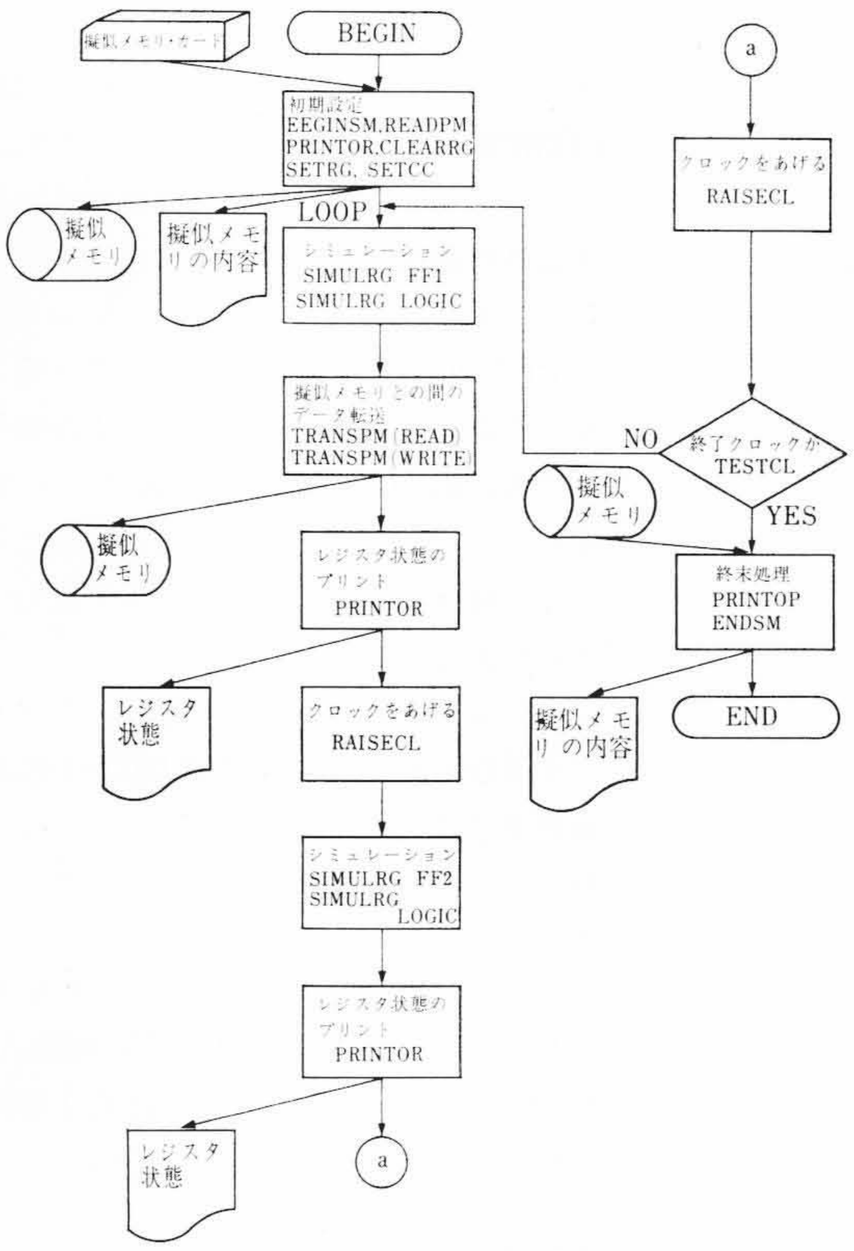


図8　シミュレーション実行制御のフローチャート例

```
TEST PROGRAM:LBS000  COMMENT: LBS TEST PROGRAM                    DATE:06.09.68. BLOCK:LOG  PAGE

BATCH NAME:BH1                MACRO NAME:PRINTRB

          C  0000000000 0000000000 0
          L  0000000000 0000000000 0
          B  0003000000 0000000000 0
REGISTER  C  0003111122 2223333444 4
NAME      K  0349025612 4783469025 6
A3(1)        .                  . .
A3(2)        .                  . X
A3(3)        .                  . X
A3(4)        .                  X .
A3(5)        .                  . .
A3(6)        .                  . X
A3(7)        .                  X X
A3(8)        .                  X .
A3(9)        .                  . .
A3(10)       .                  . X
A3(11)       .                  X .
A3(12)       .                  . X
A3(13)       .                  . X
A3(14)       .                  . X
A3(15)       .                  . .
A1(0)        ...  ..    XX    XX . .
A1(1)        ...  ..    ..    XX X X
A1(2)        ...  ..    ..    XX X X
A1(3)        .XX  ..    ..    XX . .
A1(4)        ...  XX    XX    .. X X
A1(5)        ...  ..    ..    .. . .
A1(6)        ...  XX    XX    .. . .
A1(7)        .XX  ..    ..    .. X X
A1(8)        ...  XX    XX    .. X X
A1(9)        ...  ..    ..    XX . .
A1(10)       .XX  ..    XX    XX X X
A1(11)       ...  XX    XX    XX X X
A1(12)       .XX  XX    XX    .. . .
A1(13)       ...  ..    ..    .. X X
A1(14)       ...  ..    ..    XX . .
A1(15)       .XX  XX    XX    .. X X
A1(16)       ...  XX    XX    XX . .
SCC(0)       .  X    .    X    .
SCC(1)       .  .    X    X    .
SCC(2)       .  .    .    .    X
SCC(3)       .  .    .    .    .
SCC(4)       .  .    .    .    .
SCC(5)       .  .    .    .    .
SCC(6)       .  .    .    .    .
SCC(7)       .  .    .    .    .
SCC(8)       .  .    .    .    .
SCC(9)       .  .    .    .    .
SCC(10)      .  .    .    .    .
SCC(11)      .  .    .    .    .
SCC(12)      .  .    .    .    .
SCC(13)      .  .    .    .    .
SCC(14)      .  .    .    .    .
SCC(15)      .  .    .    .    .

                                                  ONE BATCH PRINT OUT
                                                  PRINTRB JOB END
```

図10　マクロ PRINTRB によるシミュレーション結果例

```
SIMULATION PROGRAM:DATAMAKE  MACRO:READIST  COMMENT:CARD IMAGE          DATE:06.09.68. BLOCK:TRA  PAGE:  2

          HEADMIOUTRA                                                  001SIMUL
          HEADTABC TRABBB12U1MTC00001PARITY CHECK CIRCUIT TEST DATA     001SIMUL
          TABC001CAB2A1000000C0000000300000000;                        001SIMUL
          TABC0020ABCH1000000L0000000000000111111;                     001SIMUL
          TABC0030ABBC1000000000000000111111111111;                    001SIMUL
          TABC0040ABBD1000000F000001,000011100111;                     001SIMUL
          TABC0050ABBE1000000C0001,11101111111111;                     001SIMUL
          TABC0060ABBF1000000C0011111111111111111;                     001SIMUL
          TABC0070ABBZ1000000E1010101010101010101;                     001SIMUL
          ENDC                                                         001SIMUL
          ENDC                                                         001SIMUL
```

図11　初期入力状態データ例

よるシミュレーション実行制御での融通性の向上および論理記述言語による，機能ブロック・レベルのシミュレーションの実行などの新らたな試みを織り込んで所期の成果を得た。

本システムの開発にあたり，強力なご指導をいただいた，日本電信電話公社電気通信研究所，データ通信研究部，岸上調査役，高島室長，加藤調査役，交換研究部，高村補佐，都丸補佐，日本電信電話公社データ通信本部津田調査役(元交換研究部)および開発にご協力をいただいた，沖電気工業株式会社，日本電気株式会社，富士通株式会社の関係各位に厚くお礼を申し上げる。

参　考　文　献

(1)　高島，津田，加藤，戸田ほか：通研研究実用化報告 Vol. 13, No. 25, 25
(2)　高島，加藤ほか：電気通信学会交換研究会（昭和40年2月）
(3)　高島，加藤，高村ほか：情報処理，Vol. 7, No. 5, 263（昭和41年9月）