DA31－2
(1977. 1. 20)

LSI論理シミュレーションシステム；LOGOS2

黒部恒夫　根本修慈　志方洋一　可児賢二
(日本電気株式会社　集積回路事業部)

(概要)

大規模集積回路（LSI）の論理シミュレーションおよびテストテープ編集を効率良く処理するためのシステムとしてLOGOS2を作成し、実用に供してきた。以下LOGOS2システムの構成、取扱う機能単位、性能などについて概要を述べる。

1. はじめに

LSI論理回路はすでにマイクロコンピュータ、電卓用として数千ゲート／チップのものが商品化されており、数年後には1万ゲート／チップを越えるものが目標とされている。このようなLSIの論理動作は非常に複雑となっており、論理設計の確認のために論理シミュレータを用いることが定着している。LOGOS2システムは数千ゲートのLSIを対象とし、以下の各点を考慮して作成した。

① 数千ゲートの大規模回路を十分に扱えること

LSIのゲート数が増大していくと、シミュレーションにかかるCPUの増大、入力データの増大及びその出力結果の増大と繁雑化が起り、これ等はそのままシミュレーションコストと設計工数の増大に直結する。限られた期間及び費用でLSIを設計するには、これ等の増大を極力押えることが必要である。

② LSIのシステムとしての側面を考慮すること

レジスタレベルに近いモデルがゲートレベルに混在し、シミュレーション結果がレジスタレベルにも見れることが望ましい。

③ モデルが実際の回路構造からかけはなれないこと

LSI回路はレイアウト設計されてフォトマスク上に実現されるが、論理シミュレーションモデルがそれと余りにもかけはなれて対応のつけようもない状態は好ましくない。

④ デザイナーオリエンテッドであること

設計者がシミュレーションをしようと思えば、すぐ容易に行えるものであることが望ましい。膨大なドキュメントを読まねばならなかったり、シミュレーションにおよそ関係のない予備作業を必要とするものであってはならない。

⑤ LSIの設計・製造・検査の流れに組みこまれていること。

これはシステムの利用しやすさという点で重要である。従って他のプログラムシステムとのインタフェイスがとりやすいことが必要である。

2. LOGOS2システムの概要

LOGOS2システムは図1に示すように次の6個のサブシステムから構成されている。1.で述べたようにLSIの設計・製造・検査の流れに円滑に組み込まれることが考慮されている。

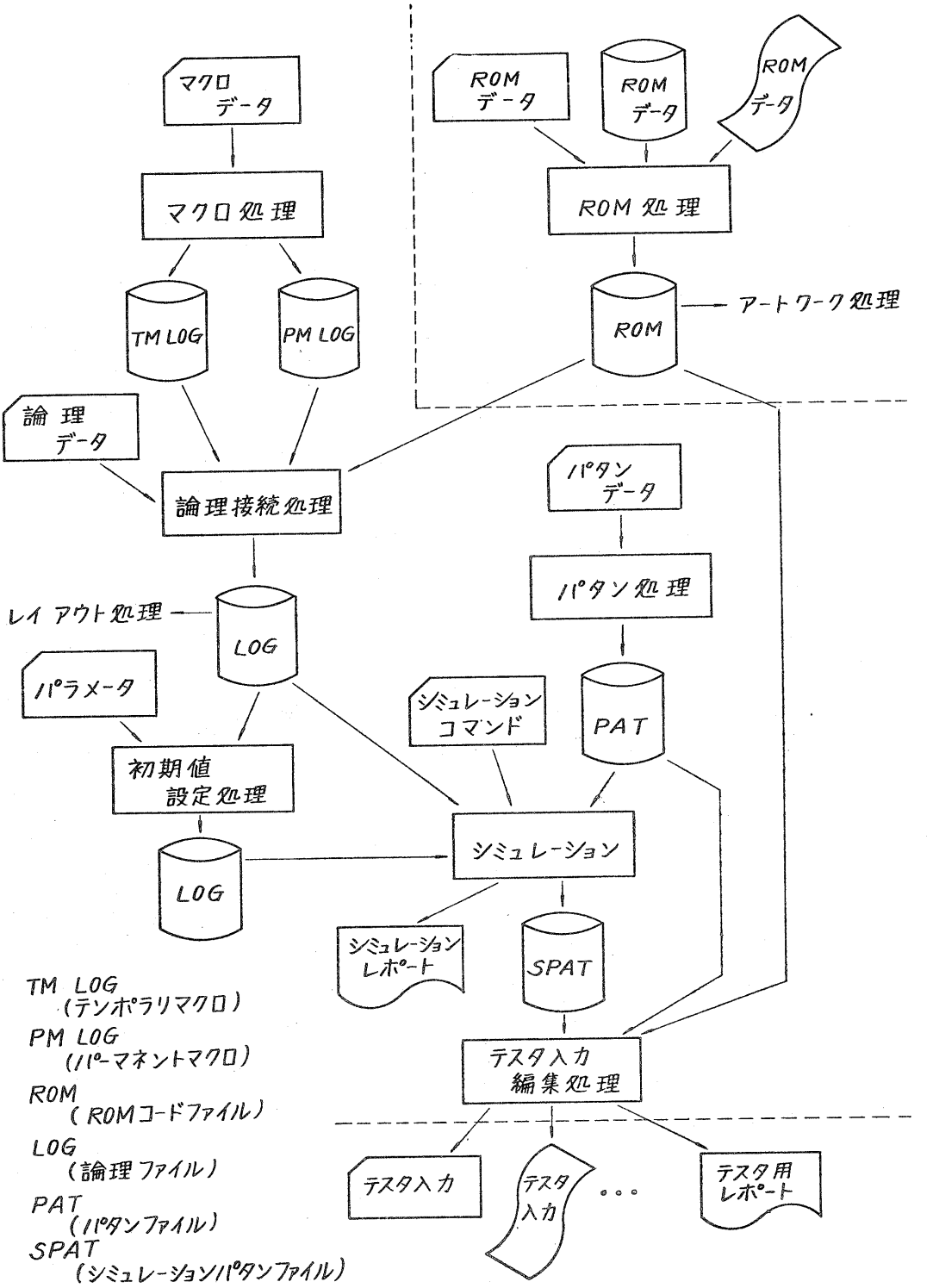


図1 LOG OS2 システムフロー

a) マクロ処理 ・・・ マクロ記述（後述）された回路をゲートレベルに展開する処理を行う。

b) 回路接続処理・・・ マクロ処理の出力ファイルであるマクロファイル、ROM処理システムの中間ファイルであるROMファイル、およびその他の回路接続記述データを読み込んでシミュレーション部分の入力ファイルであるLOGファイルを作成する。

c) パタン処理 ・・・ シミュレーション用パタン（テスト用パタンにも使う）を読み込み、シミュレーション部分の入力ファイルとなるPATファイルを作成する。PATファイルの更新ルーチンを含む。

d) 初期値設定処理・・・ LOGファイルをもとに回路状態を未定のない確定値（0又は1）に設定し新しいLOGファイルを作成する。

e) シミュレーション処理 ・・・ LOGファイル、PATファイルを入力としてシミュレーションを行う。シミュレーション結果はSPATファイルに出力する。シミュレーションはゲートレベルを基本とし、後述のようにROM、RAM，シフトレジスタはファンクションレベルで演算を行っている。シミュレーション方式は通常良く用いられる3値（0，1，X）、テーブルドリブン、単位遅延、イベント方式である。[1] 又必要に応じてシミュレーションレポート（後述）の出力も行う。

f) テスタ入力編集処理 ・・・ SPATファイル、PATファイル、ROMファイルなどをもとにして、各種LSIテスタの入力媒体（紙テープ，紙カードなど）を作成する。

3. LOGOS2で取扱う論理機能単位

LOGOS2で取扱う論理機能単位は次の3つのいずれかで記述される。

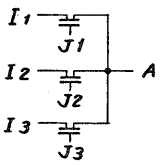
(1) 基本ゲート
(2) ファンクションブロック
(3) マクロゲート

ファンクションブロックにはROM、RAM、シフトレジスタがあり、回路記述の簡単化と，演算高速化のために 用いられる。マクロゲートは回路記述の簡単化のために用いられ，シミュレーション時にはゲートレベルに展開される。以下これらについて説明する。

3.1 基本ゲート

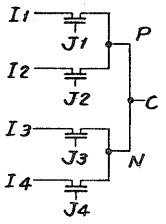
基本ゲートはAND、OR、各種フリップフロップなどで数十種がサブルーチンとして組み込まれている。特にMOSLSIではMOS特有のゲートがあり、1. で述べたようにそのモデルは実際のデバイスのふるまいに近い表現をとることが望ましい。その主なものはトランスファゲートとクロックドゲートであり、各々Pチャンネル用とNチャンネル用がある。トランスファゲートの例を図2、図3に示す。図2は3対のPチャンネルトランスファゲートの例であるが，対数は任意である。又，この例ではJ側入力の値が全て1のときは保持しているが、別に論理値X（0か1か不明である値）とするものもある。図3はCMOSトランスファゲートの例である。

又、LOGOS2システムでは単位遅延を採用しているので，全ゲートに等しい単位遅延が与えられるが，ユーザが特に遅延を与えたいときは遅延ゲートで遅延値を与えることができる。



J1	J2	J3	A
0	1	1	I1
1	0	1	I2
1	1	0	I3
1	1	1	保持
0	0	0	I1+I2+I3
0	0	1	I1+I2
0	1	0	I1+I3
1	0	0	I2+I3

図2 PチャンネルMOSトランスファゲート



$P = I_1\overline{J_1} + I_2\cdot\overline{J_2}$
(但し.P≠1の時はP=Z)

$N = \overline{\overline{I_2}\cdot J_3 + \overline{I_4}\cdot J_4}$
(但し.N≠0の時はN=Z)

P	N	C
Z	0	0
1	Z	1
他		X

(Zは論理的に無視すべき値)

図3 CMOSトランスファゲート

3.2 ファンクションブロック

先に指摘したように，ゲート数が増えてくるとCPU時間は比例以上の勾配で増え，回路規模が1万ゲートにもおよぶと，シミュレーションすること自体が禁止的になることが予想される。このような状態を緩和するためには，回路モデルにゲートより大きい単位のものを導入して，シミュレーション対象素子数を減らすことが必要となる。そこでLOGOS2システムではレジスタレベルに近いモデルとしてファンクションブロックを導入している。これはマクロのようなユーザが定義可能ものとはしないで，システム内の組込み関数として定義し，より速いスピードで演算できるようにしている。当然のことながらファンクションブロックを導入することによって回路記述も簡単化された。

・ROM

ROMは図4に示されるように、クロックとアドレス入力($A_1, A_2, \cdots, A_r$)によって表現される。クロックがオンの時に$A_1, A_2, \cdots, A_r$で示されるアドレスのROMデータが出力($O_1, O_2, \cdots, O_m$)へ出される。ROMのデータは図1に見られるように、ROM処理用のプログラムの出力を利用することができる。それができないときは何番地にどのデータが対応するという簡単な指定を前もってしておけばよい。

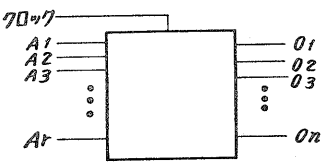


図4 ROM

・RAM

RAMは図5に示されるように、書き込み用クロック(W)、読み出し用クロック(R)、データ入力(D)とアドレス入力(A)により表現される。書き込み用クロックがオンの時にはアドレス入力によって示されるアドレスにデータ入力の値がメモリされる。読み出し用クロックがオンのときにはアドレス入力により示されるアドレスに記憶されているデータが出力される。

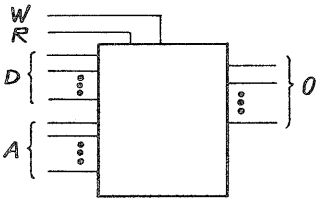


図5 RAM

・ダイナミックシフトレジスタ

図6のようなクロック C_1 と C_2 によって制御されるダイナミックシフトレジスタはそのデータ入力とクロック C_1, C_2 およびシフト段数によって記述される。 演算は各シフト段ごとにバッファーを持たせて，クロック C_1 がオンのときに前段出力をバッファーに取り込み、C_2 オンのときにバッファー値を出力する。これを全シフトに対して同時に行う。

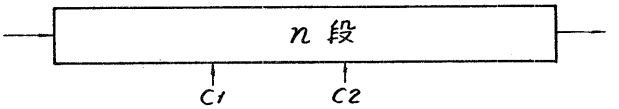


図6 シフト・レジスタ

3.3 マクロゲート

マクロゲート（以下マクロという）は主として、基本ゲートやファンクションブロックそのものではないが，回路内に何度も現われるような同一のブロック（ゲートの集り）の記述を容易にするために用いられる。実際に約1,200ゲートの回路が、マクロを用いて300行程度の論理記述におさまった例がある。また、回路を適当な機能の単位にマクロでまとめて記述すれば非常に理解しやすくなる。

マクロは基本ゲートやファンクションブロック（メモリ系を除く）と既登録のマクロの組合せで記述される。マクロのネスティングに制限はないが，自分自身を参照することは許されない。

マクロにはテンポラリとパーマネントの二種があり，パーマネントは一度登録すると次回以降は参照するだけでよい。テンポラリはランの度に消去される。主として一般性のあるマクロがパーマネントとして登録される。

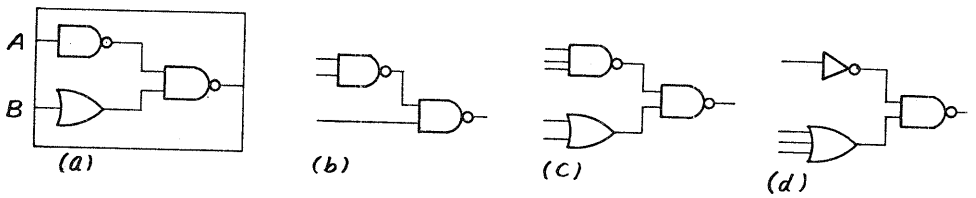


図7 マクロゲート

マクロは通常等価なブロックの記述を簡略化するために用いられるが，完全に等価ではないがほぼ同じであるようなブロックも扱えるように工夫してある。図7(a)のマクロはAとBは各々1入力であるが、これを群入力として定義すれば、AとBは各々1以上の入力を代表していることになり、図7(b)(c)(d)で示されるブロックは全てこのマクロにより記述できる。

マクロは更に、ある一定の規則を持ったブロックの繰り返しの記述を簡略化できる。例えば、図8のようなマクロの連鎖において、個々のマクロの出力が次のマクロへ入力し（入力する位置は全マクロ同一とする。）各マクロの他の入力が共通であるようなものは、最初のマクロの記述と繰り返しの数を指示するだけでよい。

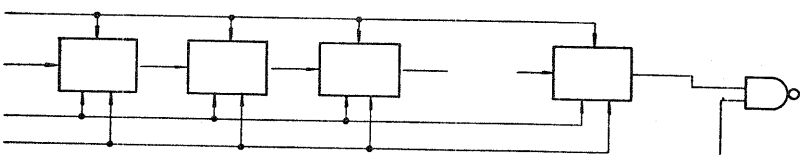


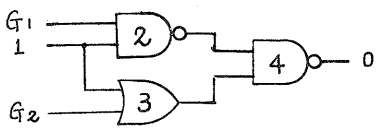
図8 マクロの連鎖

3.4 回路記述例

図9にLOGO2の回路接続の記述例を示した。1.で述べたように入力記述は非常に簡単で初心者が簡単なマニュアルを読んですぐに使えるようになっている。

図9(a)はマクロの記述例であり，SELとNAD2がその名前である。右に書かれている回路がSELであり，G_1とG_2は群入力となっている。　そのSELを用いた回路記述の例（一部分）が図9(b)である。群入力のきれめを定義するためにカンマ(,)を用いている。（回路記述は固定カラムのfan-in表示形式）

```
   L O G I N C   C A R D   I M A G E   L I S T

          1         2         3         4         5         6         7
   1...*....0....*....0....*....0....*....0....*....0....*....0....*....0....*..
 1 *LOGINC    KAIRO
 2 /MACRO     SEL
 3   G1       GIN
 4   G2       GIN
 5   1        JIN
 6   2        NAND     G1          1
 7   3        OR       G2          1
 8   4        NAND     2           3
 9   0        JOUT     4
10 /END
11 /MACRO     NAD2
12   G1       GIN
13   G2       GIN
14   1        AND      G1
15   2        AND      G2
16   3        NOR      1           2
17   0        JOUT     3
18 /END
```



SEL

(a) マクロ例

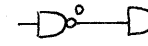
```
*****    LOGINC   PROGRAM   RUNNING   LIST    *****     CIRCUIT NAME : KAIRO

   L O G I N C   C A R D   I M A G E   L I S T

          1         2         3         4         5         6         7
   1...*....0....*....0....*....0....*....0....*....0....*....0....*....0....*..
51   A007     NAND     IRTP
52   A037     NAND     A002
53   A009     NAND     P3
54   A198     NAND     A009       A008
55   A102     NAND     A009
56   A197     NAND     A009       A037
57   A038     NAND     LIST
58   A041A    NOR      CHECK      A038
59   A041     NOR      A041A      A040
60   A040     NOR      A038       A037       A102
61   CL2N     OR       CLK        TEST
62   CL1N     OR       CLK        TEST1
63   A042     NOR      F044   01  CL1N
64   A043     NOR      F045   02  CL2N
65   F044     FFR      A041      ,A008
66   F045     FFR      A042      ,A008
67   S01      SEL      EN0       ,A041A     ,A038
68   S02      SEL      EN0        EN1       ,A041      ,A038
69   S03      SEL      EN1        EN3       ,A041      ,A038
70   S04      SEL      EN2       ,EN4        EN5       ,F044   01
71   S05      SEL      EN6       ,EN0       ,A042
72   S06      SEL      EN3        EN7       ,A043      ,A042
73   S07      SEL      EN7       ,EN2        EN5       ,A043
74   S08      SEL      EN4        EN5       ,EN0       ,A043
```

(b) マクロ使用例

図9　回路記述例

4. 初期状態の設定

ＬＯＧＯＳ２システムでは回路の初期状態を特に指定がない限り Ｘとしている。普通順序回路に対しては、フリップフロップなどの値を設定するための初期値設定シミュレーションが行なわれたあとで，機能確認のシミュレーションが行なわれる。そこでＬＯＧＯＳ２システムではこのようなときに、初期セットが成功したか否かをチェックして失敗した場合には全状態をダンプできるようにしている。

回路によっては、回路の初期状態として動作開始時（電源入力時）の状態をそのまゝ使うことがある。このような場合、人手によって初期値を与えたり，初期設定用の論理とパタンを追加することが良く行なわれている。ＬＯＧＯＳ２でもこの様な方法をとることができる。しかし，この方法は手間がかゝり，余分な論理をつけ加えねばならず好ましくない。又、人手で初期値を与えたときは与えるが不十分なためシミュレーション中にその値が無効になることがある。又、設定値に矛盾がある（例 

は矛盾している）ときは、シミュレーションが不正確になるといった問題がある。　このような場合には回路内の状態が矛盾のない確定値（０又は１）に安定していれば問題はない。そこで、ＬＯＧＯＳ２では回路の状態を矛盾なく自動設定する初期値自動設定ルーチンを設けてこれを解決している。

5. シミュレーション時間

ＬＯＧＯＳ２システムは現在ＡＣＯＳシステム７００で稼動中（表１の例は全てメモリ４６Ｋワード以内）であり、使用言語はＦＯＲＴＲＡＮとアセンブラでプログラム実行文は約１０Ｋステップである。

表１にランタイムの例を示す。　１イベントあたりのＣＰＵ時間は約 0.38m sec である。

	ゲート数	ROM 有無	パタン数	シミュレーション時間
1	950	なし	3,000	2分 21秒
2	800	384 ワード	2,000	1分 3秒
3	1,400	なし	3,600	1分 16秒
4	400	なし	4,700	39秒
5	800	なし	1,000	35秒
6	1,200	1,024 ワード	1,700	1分 56秒
7	750	なし	500	14秒

表１　シミュレーション時間例

6. シミュレーションレポート

回路が大規模化するにしたがって、一般にシミュレーションのテストケースが増え、それにともなってシミュレータから出力されるレポート量が増大し、その解析作業が大仕事となってくる。LOGOS2システムではシミュレーション結果の出力をコントロールするコマンドを豊富に用意し、それを用いることによりレポートを簡略化し解析作業の省力化を図っている。

主な機能を以下にあげる

(1) シミュレーションの結果として、入出力のパタンがリスト上にレポートされるが、全パタン必要のないときは必要なところだけを指示することができる。指示はどこからどこまでという区間の指定や、何パタンおきという繰り返しの指定などを組合せて任意にできる。

(2) あらかじめ予想出力パタンを入力しておくことにより、結果を自動的にチェックすることができる。どのパタンのどの出力ピンで異なったかは、入出力パタンのリスト上で1目で識別できるようにしてある。

(3) 入出力パタンはタイムチャート処理によって、タイムチャートとしてもプロットできる。図10に例を示した。

(4) 回路の内部状態のダンプも入出力と同じようにレポートされる。たゞし、どの素子をダンプするかはユーザ自身が指示しなければならない。

レジスタなどのようにまとめてダンプしたろがわかり易いものはグループ化してダンプでき、表示も16進、8進、4進、2進のどれでも任意にえらぶことができる。

(5) 全状態のダンプが必要なときには、その必要であるパタン番号を指示することにより可能となる。

(6) オシレーションを起こしたときなどの解析のために、遷移状態をトレースすることができる。

7. おわりに

LOGOS2の主な改良点として次の3つが挙げられる。

(1) タイミング解析

(2) 会話型の利用

(3) ハイレベルモデルの取扱い[2]

文献

(1) 例えば M.A. Breuer (Ed.), "Design Automation of Digital Systems, vol. 1 – Theory and Techniques", Prentice-Hall, Englewood Cliffs (1972).

(2) 例えば M.A. Breuer (Ed.), "Digital System Design Automation : Languages, Simulation & Data Base", Computer Science Press, Inc., Woodland Hills, California (1975)

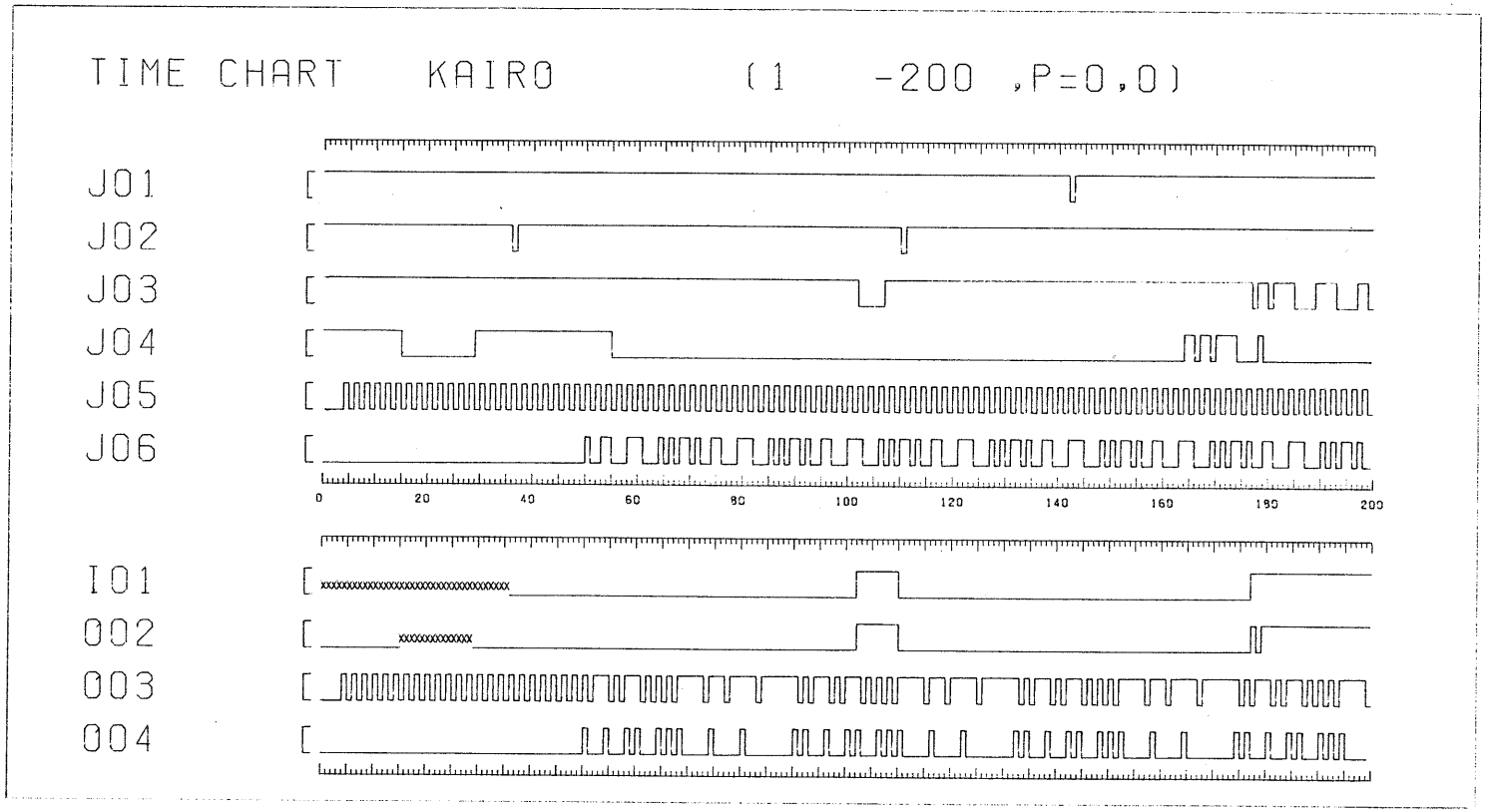


図10 タイムチャート例