非同期シミュレータの開発

豊島哲男，伊藤　誠，蒸沼良一
山梨大学　工学部　電子工学科

1．はじめに

本研究室では論理設計自動化システム，実装自動化システムを開発し，実用しているが，さらに，論理設計の誤りの検出のためにシミュレータを開発したので報告する．シミュレータの対象言語は，ICの機能単位に記述するモジュール表現である．シミュレーションは3値で行ない，結果はラインプリンタで印刷する．個々の素子（モジュール）のシミュレーションは対応するサブルーチン（以後インタプルタルーチン）の呼出しで行われるから，サブルーチンの追加により，シミュレーションの対象となる素子を拡張できる．また，インタプリンタルーチンは，論理値以外に一般の整数値が許されるから，ワンショット・遅延素子もシミュレーションできる．ただし，CR（コンデンサ／抵抗）による積分回路，素子の負荷による動特性の変化等は考慮していない．

2．設計自動化システムの概要

本シミュレータは，研究室で開発している設計自動化システムの一部分を構成しているので，まず，このシステムの概略を紹介する．設計する回路はまず，ムDRANと呼ぶ言語で記述され，コンバータによりモジュール表現に変換される．モジュール表現は回路図のレベルに対応し，直接，モジュール表現で記述することもできる．これは，実装システムにより，実装情報と照合されてICの割当・配置決定後，結線表に変換される。

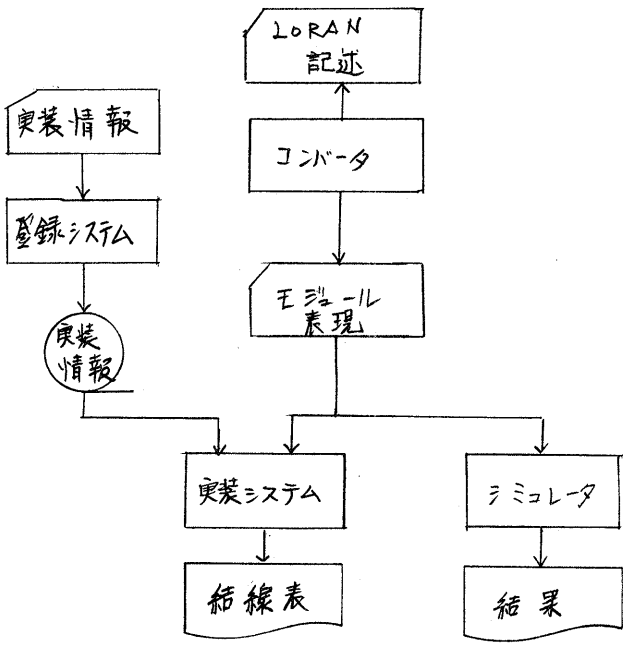


図 2.1　設計自動化システム

この結線表により，ワイヤ・ラッパによる配線を行う．このシステムは現在実用化され，マイクロCPU2台，PTR，LP，簡易ディスクをもった小型計算機システム－エミュレータの開発に利用された。本シミュレータは，実装設計前に，論理設計のデバッグに利用するものである．システム全体は，IC数50程度の汎用基盤を対象としている。コンバータ

はすべて，ＦＯＲＴＲＡＮで記述しており，バッチモニタの下で動作する。したがって，全行程ともトップダウン形式であり，設計者は介入できない。また，製作も一品料理になることが多いため，プリントパターンの生成は考えていない。ただし，新らしい機能をもったＩＣがすぐに利用できるように，全システムを構成している。現在システムに登録されているＩＣの種類数は４０程度であり，ＩＣの追加は実装情報の登録および，シミュレータルーチンの追加のみで可能となる。

３、素子のモデル

3.1 素子の動特性

素子はｍ入力ｎ出力（図３.１）であり，入力には非論理値が許される。出力の変化は，入力論理の変化によってのみ起り，その変化は最小遅速時間と最大遅速時間の間に起るものとする。（図３.２）最小と最大の遅速時間の間は，論理値は確定していないと考え，〝２〟で表わす。したがって，論理値は一般に０→２→１あるいは，１→２→０と変化する。ここでは出力論理値の値が確定するのは，入力論理値の確定後，最大遅速時間経過した時と考える。そのため，素子を通るごとに一般に論理値２の区間が増大する。インバータゲートの入出力特性を図３.２に示す。遅速時間は，素子および出力線によって別々の定義することができる。

3.2 素子の表現

ここでは，図３.１の素子を

Ｍ（Ｉ１，Ｉ２，…… Ｉｍ ／Ｏ１，Ｏ２ …… Ｏｎ）

で表現し，これをモジュール文と呼ぶ．ここにＭは，論理機能をあらわす名前である。また，接続関係は信号線の名前の対応で示す．図３.３にその例を示す．名前の〝－〟記号は通常負論理を示す。フリップ・フロップ（ＦＦ），カウンタ等の場合モジュール文内の変数の位置がその変数（信号線）の機能を決定する．ＪＫ型マスタ・スレーブＦＦの例を図３.４に示す．モジュール文の集合をモジュール表現と呼ぶ．

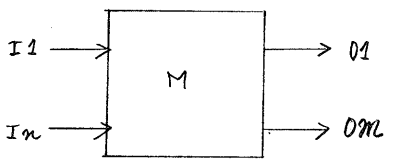
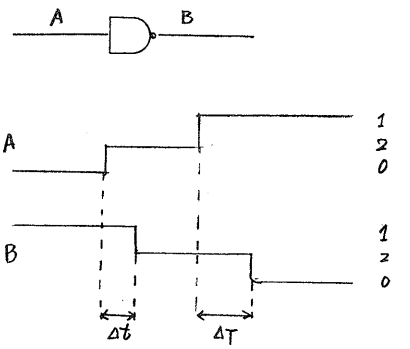
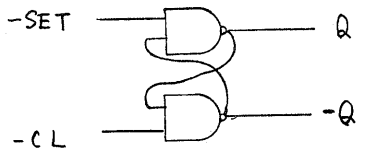


図３.１　素子のモデル



Δt　最小遅速時間
ΔT　最大遅速時間

図３.２　インバータの動作

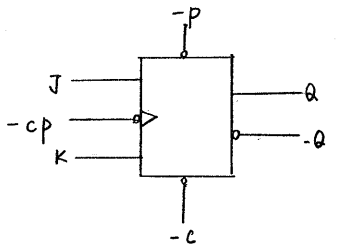


NAND2(-SET, -Q / Q)

NAND2(Q, -CL / -Q)

(a) (b)

図 3.3 モジュール表現



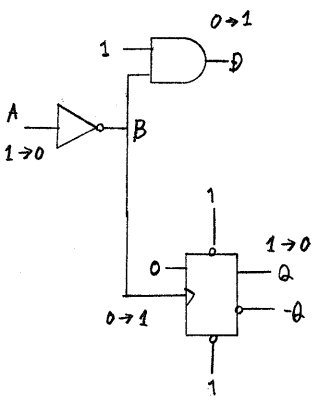
FF(J, K, -cp, -P, -C / Q, -Q)

(a) (b)

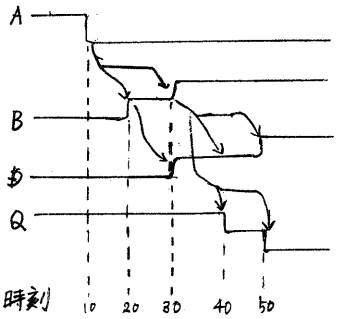
図 3.4 フリップ・フロップの表現

4. シミュレーションの原理.

回路中の信号変化をここではイベントと呼ぶ, シミュレータは, まず, 外部よりイベントを読みとる. (イベントの生成). このイベントはその信号を入力とする素子に入り(場合によっては)その出力を変化させることにより, 新らしいイ



(a) 例の回路



(b) タイム チャート

時刻	信号名	変化
10	A	1→0
20	B	0→2
30	B	2→1
	D	0→2
50	Q	1→2
	-Q	0→2
70	D	2→1
	Q	2→0
	-Q	2→1

(c) イベント テーブル

図 4.1 シミュレーションの原理

ベントを生成する。イベントは変化する時刻，変化する信号（変数）の名前，変化のしかた，の3つの要素から構成される。シミュレータは生成されたイベントを時刻の順にソートしてイベントテーブルに記憶する。

以下図4.1を例にして説明する。各素子の最小遅延および最大遅延時間は，10および20とする。シミュレーションの時刻が10となると，まず，時刻10のイベントをイベントテーブルより取りだす。イベントは，変数Bが1→0に変化することを示している。シミュレータは，Aの値を0に変化させた後，Aを入力とする素子（モジュール）をさがし，新らしいイベントの生成を試みる。この場合，インバータが動作し，時刻20で0→2，30で2→1，の2つのイベントが生成される。（(b)のタイムチャート参照）シミュレータはこの2つのイベントをテーブルに登録し，時刻を進める。シミュレーション時刻が20になると，Bの0→2の変化が実行され，この変化は2入力ANDの出力を変化させ，時刻30で信号Dが0→2に変化するイベントを生成する。すなわち，(b)の論理変化が，イベントテーブル上の表の形で登録されていく。したがって，シミュレータは基本的には，次のフローを実行すればよい。

① 外部指示によるイベントの登録
② イベントのとりだしと実行
③ 新らしいイベントの生成と登録
④ 時刻を1つ進めて②へ戻る

5. シミュレータの開発

5.1 入力形式

入力は，図5.1のように記述する。▽MODULE以下に対象となる回路を記述する。▽INTERVALから▽INITIAL SETはシミュレータへの制御情報であり，記述順序は任意である。▽OUTPUTは▽PRINT TIME毎に印刷する出力変数を指定する。▽VALUE SETは非論理値の値を設定し，▽INITIAL SETは論理値の値を設定する。ここで設定されない全変数は論理値2を初期値とする。<外部イベント>は<時刻><論理値列>で記述される。これは<時刻>で指定された時刻に，▽INPUTで指定された信号が<論理値列>に対応する値に変化することを指定する。<外部イベント>はシミュレーション時に順に読込れるから，<時刻>でソートして入力する必要がある。

```
▽MODULE
    <モジュール表現>
▽INTERVAL
    <シミュレーションの単位時間>
▽STIME
    <シミュレーションの打切時間>
▽PRINT TIME
    <印刷時間間隔>
▽INPUT
    <入力変数列>
▽OUTPUT
    <出力変数列>
▽VALUE SET
  {<非論理変数>=<正整数>}+
▽INITIAL SET
  {<論理変数>=<論理値>}+
▽END
    <外部イベント>+
```

図5.1 入力の記述形式

5.2 データ・ベース（テーブル形式）

データ・ベースはすべてテーブル形式をしている。まずモジュール表現を記憶するのがMDT（MoDul Table），VNT（Variable Name Table），CNT（CoNection Table）である。MDTは定義されているモジュール名とその入出力変数の数を記憶する。VNTは変数名とその値およびCNTのポインタから構成され，CNTはモジュール表現をMDTとVNTへのポインタをつけて記憶する。

イベントは，TMPとETLからなる．TMP（Time MaP）は一次元配列でi番目の要素が，i番目の時刻のイベントを記憶する．イベントはすべてETLにリスト形式で記憶され，TMPはその先頭要素へリンクする．TMPがつきたとき，先頭に戻り，ETLがつきたときはガベージコレクタが自動的に働く。ETLが未処理イベントでオーバーフローすると，エラーメッセージと共にシミュレーションを打切る。TMPおよびETLのサイズはプログラム中の変数で変更されるが，コンパイルをし直す必要がある。

その他，入出力変数列を記憶するIPT，PRT，インタプリタへのパラメータとなるINVAR，OUTVAR，DLYのテーブルがある。後者については，後に説明を加える。

5.3 シミュレーションアルゴリズム

シミュレーションの概略はすでに述べた通りである。ここでは，もう少し細かい点について議論する。

a) 論理値の初期化

通常すべての論理値は0か1か不明である。したがって，特に指定がない限り初期値は2とする。しかしながらワンショットの出力および0か1に固定してある論理は最初から定まっている。この場合は，'INITAL SETの形で論理値の指定が必要となる。この指定は，時刻0でのイベントとして登録される。したがって，初期指定された値による間接的な論理値決定はシミュレーション実行段階で定まることになる。そのため，外部入力を変化させる時刻は，初期値によるイベントが消失するところにする必要がある。

b) シミュレーションの打切り

現在は，'STIMEにより打ち切り時間でのみシミュレーションを打ち切っている。ただし，ムダな印刷が多いので，以後イベントが存在しなくなった時点でも打ち切るようにしたい。

c) モジュールインタプリタの呼出し

基本的には各イベントに対し，それを受けるモジュールを呼び出す必要がある。ただし，同一時刻に同じ素子に対するモジュールインタプリタを呼び出さないように注意している。また，イベントの動く方向が現在の論理値(0, 1)になっている場合もインタプリタは呼び出さない。

d) 外部イベントの読込

外部イベントは一つ先読をする形式にしている。すなわち，登録された外部イベントの処理をした時点で次のイベントが読込まれる。

e) モジュールの追加

入力でA（−A）があり，出力には−A（A）しかないとき，論理反転モジュールINV（−A／A）（INV（A／−A））を追加する。また，Wired ORに対して，適切なANDモジュールの追加をするようにしたい。

f) プログラムのモジュール化

プログラムは，できるだけ汎用型のサブルーチンに分割している．例えば，リスト型式のイベントの記憶呼び出しおよび，ゴミ集め等は，別途開発した汎用処理ルーチンを利用している．また，各種テーブルの更新，管理もすべて汎用ルーチンを用いている．現在，サブルーチン数は，モジュール インタプリタ ルーチンを除いて30程度である．ただし，汎用ルーチンの使用がシステムの効率（処理速度／サイズ）を悪くしていることも否定できない．

5.4 モジュールインタプリタの記述

各モジュールに対し，一つのモジュール・インタプリタと称するサブルーチンが必要である．シミュレータは，FORTRANのコモン領域を通して入力の値を引渡すから，インタプリタはそれより出力を計算して，出力論理値および遅延時間を決定して，同じくコモン領域を通して，シミュレータ本体に結果を送る．フリップフロップのトグル（クロック）端子のように論理値の変化情報が必要な場合があるので，入力はイベント前後の論理値を別のテーブルに記憶して引渡している．2入力ANDゲートの例を図5.2に示す．

モジュール・インタプリタは入出力の値をパラメータとしているから，外部にあらわれないメモリを持つモジュールの記述はできない．例えば，マスタ・スレーブ型FFはマスタ側の出力が外に出ていないので，正確な記述はできない．現在は，立下りのエッジトリガ型で代用している．また，複数個のメモリモジュールも記述できない．このような場合，メモリモジュールに番号をつけて記述すればシミュレーション可能となるが，実装システムとコンパチブルでなくなる．

また，このルーチンはシミュレーション時間を大きく左右してしまう．したがって，大きな回路を対象とするシミュレータに，この方式を採用することは困難であろう．

```
      SUBROUTINE AND2
      COMMON /INPTV/AINVAR(20), NINVAR(20)
     1       /OTPV/ AOTVAR(20), NOTVAR(20)
     2       /DLY/ DMIN(20), DMAX(20)
C     ** MODULE WA AND2(I1, I2/IOUT) **
      EQUIVALENCE (I1,NINVAR(1)),(I2,NINVAR(2)),(IOUT,NOTVAR(1))
      IOUT=I1*I2
      IF(IOUT.GT.2) IOUT=2
      DMIN(1)=10.0E-9
      DMAX(1)=22.0E-9
      RETURN
      END
```

図 5.2 2入力 AND のインタプリタ

5.5 シミュレータのフロー

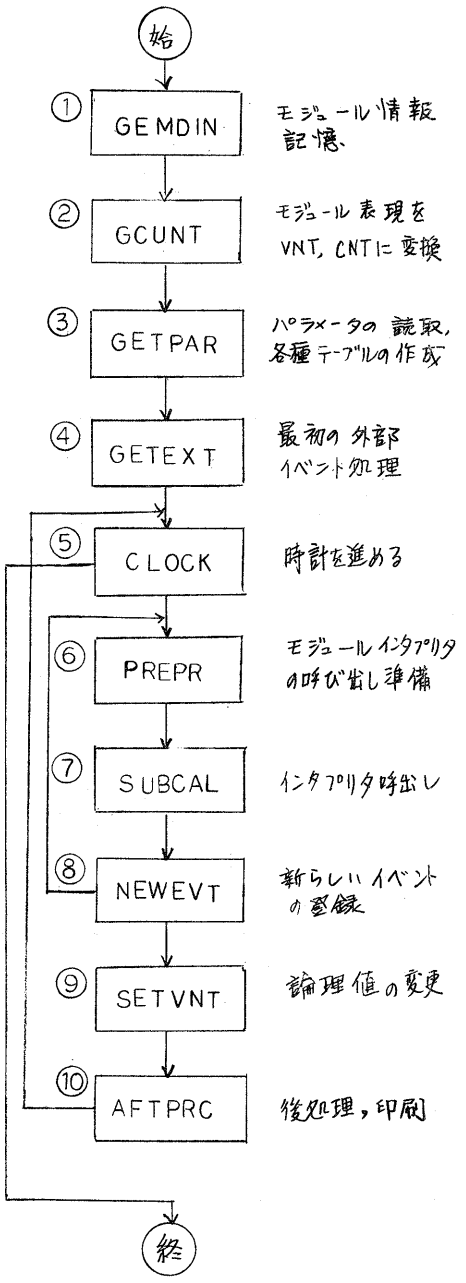


図 5.3 シミュレータのメインプログラム

シミュレータのメイン・プログラムのフローは、図5.3のようになる。①へ③はシミュレーションの準備，④～⑩が実行部分であり，両者はオーバレイして実行させている。

シミュレーションはすべてFORTRAN（インタプリタを除いて2000ステップ）で書かれ，各種テーブルはすべて主記憶常駐である。実行速度はIC数30程度で、2～3分である。（FACOM 230-45S）

6 例

図の6.1の簡単な入力と出力の具体例を示す。CPU処理時間は4秒である。

7 おわりに

このシミュレータはIC数で50程度の回路を対象にしている。したがって，各種テーブルはすべて主記憶在中であり，それだけ，開発は容易であった。

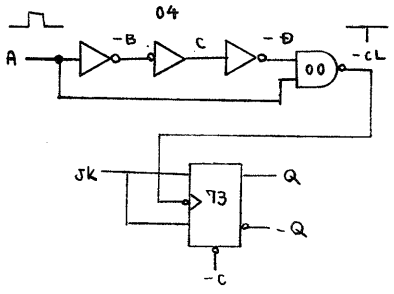
今後の問題としては①ワンショット等，固定した出力を持つ信号に対しては，対応する初期値を自動的に与える，②〈外部イベントの記述を簡単にする，③故障検査への利用を考える，④〈プログラムの全面書換になるが〉小型CPUとディスプレイ端末を利用して，インタラクティブなシミュレータ/チェッカーを作成する，ことを考えている。

```
NOLIST
C  **** 3STAGE DELAY ****
INV(A/-B)
INV(-B/C)
INV(C/-D)
NAND2(A,-D/-CL)
FFC(JK,JK,-CL,-C/Q,-Q)
END
```

モジュール表現

```
    **CONTROL INFORMATION**
INTERVAL 0.500E-08
SIMULATION TIME 0.300E-05
PRINT TIME 0.100E-07
INPUT A,-C
OUTPUT A,-B,C,-D,-CL,-C,Q
**INITIAL SET**
JK=1    END
```

パラメータ



回路図

図 6.1 例

TIME	A	-B	C	-D	-CL	-C	Q
0.450E-06	*	*	*	*	*	*	*
0.465E-06	*	*	*	*	*	*	*
0.480E-06	*	*	*	*	*	*	*
0.495E-06	*	*	*	*	*	*	*
0.510E-06	0	*	*	*	*	0	*
0.525E-06	0	1	*	*	1	0	0
0.540E-06	0	1	*	*	1	0	0
0.555E-06	0	1	0	*	1	0	0
0.570E-06	0	1	0	*	1	0	0
0.585E-06	0	1	0	1	1	0	0
0.600E-06	0	1	0	1	1	1	0
0.615E-06	0	1	0	1	1	1	0
0.630E-06	0	1	0	1	1	1	0
0.645E-06	0	1	0	1	1	1	0
0.660E-06	0	1	0	1	1	1	0
0.675E-06	0	1	0	1	1	1	0
0.690E-06	0	1	0	1	1	1	0
0.705E-06	0	1	0	1	1	1	0
0.720E-06	0	1	0	1	1	1	0
0.735E-06	0	1	0	1	1	1	0
0.750E-06	0	1	0	1	1	1	0
0.765E-06	0	1	0	1	1	1	0
0.780E-06	0	1	0	1	1	1	0
0.795E-06	0	1	0	1	1	1	0
0.114E-05	0	1	0	1	1	1	0
0.116E-05	0	1	0	1	1	1	0
0.117E-05	0	1	0	1	1	1	0
0.118E-05	0	1	0	1	1	1	0
0.120E-05	0	1	0	1	1	1	0
0.121E-05	0	1	0	1	1	1	0
0.123E-05	0	1	0	1	1	1	0
0.125E-05	0	1	0	1	1	1	0
0.126E-05	0	1	0	1	1	1	0
0.127E-05	0	1	0	1	1	1	0
0.129E-05	0	1	0	1	1	1	0
0.130E-05	1	1	0	1	1	1	0
0.132E-05	1	*	0	1	*	1	0
0.133E-05	1	0	*	1	0	1	*
0.135E-05	1	0	1	*	0	1	1
0.136E-05	1	0	1	*	*	1	1
0.138E-05	1	0	1	0	*	1	1
0.139E-05	1	0	1	0	*	1	1
0.141E-05	1	0	1	0	1	1	1
0.142E-05	1	0	1	0	1	1	1
0.144E-05	1	0	1	0	1	1	1
0.145E-05	1	0	1	0	1	1	1
0.147E-05	1	0	1	0	1	1	1
0.148E-05	1	0	1	0	1	1	1
0.150E-05	1	0	1	0	1	1	1
0.151E-05	1	0	1	0	1	1	1
0.153E-05	1	0	1	0	1	1	1
0.154E-05	1	0	1	0	1	1	1
0.156E-05	1	0	1	0	1	1	1
0.157E-05	1	0	1	0	1	1	1
0.159E-05	1	0	1	0	1	1	1
0.160E-05	0	0	1	0	1	1	1
0.162E-05	0	*	1	0	1	1	1
0.163E-05	0	1	*	0	1	1	1
0.165E-05	0	1	*	*	1	1	1
0.167E-05	0	1	0	*	1	1	1
0.168E-05	0	1	0	1	1	1	1
0.169E-05	0	1	0	1	1	1	1
0.171E-05	0	1	0	1	1	1	1
0.172E-05	0	1	0	1	1	1	1
0.174E-05	0	1	0	1	1	1	1
0.176E-05	0	1	0	1	1	1	1

結果 →