論理回路の正確なタイミング検証のための時間記号シミュレーション†

石浦 菜岐佐†† 高橋 瑞樹†† 矢島 脩三††

論理回路のタイミング検証の一手法として，時間記号シミュレーションという新たな手法を提案する．従来から提案されている論理回路の記号シミュレーションが，信号線の信号値を記号として扱うのに対し，時間記号シミュレーションは時間を記号として扱う．すなわち，ゲート遅延や入力の変化する時刻を変数で表現することによってそのばらつきをモデル化し，シミュレーション時刻をこの変数を含む式のままで扱うことにより，タイミングの解析を行おうとするものである．本論文では組合せ回路向きの効率的なアルゴリズムとして，時間優先評価アルゴリズム（Tアルゴリズム）に基づく計算法を示す．時刻を表す式の簡単化や比較は，シミュレーション中に現れる式が一次式に限られることに注目し，線型計画法によって処理する．時間記号シミュレーションを用いれば，最大／最小遅延論理シミュレーションで問題となっていた悲観性のない正確な結果が得られるため，ハザードなどによる誤動作の有無を正確に判定することができる．さらに，時間記号シミュレーションでは，シミュレーション結果から直ちに誤動作が起こる条件やクリティカル・パスを求めることができるため，誤動作の原因追跡や設計の改善にも利用することができる．

1. はじめに

近年の集積回路技術の進歩に伴い，より大規模で高機能なディジタル・システムが実現されるようになっているが，その設計検証をいかに行うかという問題が，計算機援用設計技術に課せられた大きな課題となっている．なかでもタイミングに関する検証は複雑で困難な場合が多く，大規模な論理回路はタイミング検証の容易な同期式順序回路として設計されることが多い．しかし，通信制御などの周辺回路や記憶素子など，非同期回路として設計される部分に対しては，仕様どおりの動作をするかどうか，ハザード，発振，競合などによる誤動作が生じないかどうかを調べるために，論理的動作との連携を考慮した精密なタイミング検証が必要になる．このように微妙なタイミングが問題になる場合には，製造条件や使用環境の変化による素子遅延のばらつきまでを考慮することが必要になる[10]．

現在最も広く用いられている設計検証の手段である論理シミュレーションで，このような遅延のばらつきを扱う方法としては，最大／最小遅延モデルによるシミュレーションが提案されている[1]．しかし，このモデルによるシミュレーションは，結果があまりにも悲観的になる，すなわち，正しい設計に対しても誤動作の可能性を示す信号値が大量に出力されることが多く[1]，このため，真に誤りがあるのかどうか，あるいはその原因がどこにあるのかを解析することが極めて難しい，という問題が従来から指摘されてきた[10]．

本論文ではこのような問題を解決する一手法として，時間記号シミュレーションという新たな論理シミュレーションの手法を提案する．従来から提案されている記号シミュレーション[2]が，信号線の信号値を記号として扱うのに対し，時間記号シミュレーションは時間を記号として扱う．すなわち，ゲート遅延や入力の変化する時刻を変数で表現することによって，そのばらつきをモデル化し，シミュレーション時刻をこの変数を含む式のままで扱うことにより，正確なタイミング解析を行おうとするものである．シミュレーション時刻が定数でないため，従来のシミュレーション・アルゴリズムでは計算が困難であるが，対象回路を組合せ回路に限定し，時間優先評価アルゴリズム（Tアルゴリズム）[3]の考え方に基づけば，比較的効率よくシミュレーションを行うことが可能になる[5),6)]．時刻を表す式の簡単化や比較は，一般には難しい問題であるが，本シミュレーションにおいては式の形が変数の一次式に限られることに注目し，線型計画法に帰着して処理する[5]．

時間記号シミュレーションは，イベントの因果関係を考慮して計算を行うシミュレーション手法に属し，最大／最小遅延論理シミュレーションで問題となっていた悲観性のない正確な結果が得られるため，ハザードなどによる誤動作の有無を正確に判定することがで

† Time-Symbolic Simulation for Accurate Timing Verification of Logic Circuits by NAGISA ISHIURA, MIZUKI TAKAHASHI and SHUZO YAJIMA (Department of Information Science, Faculty of Engineering, Kyoto University).

†† 京都大学工学部情報工学教室

きる．のみならず，シミュレーション結果から直ちに，誤動作が起こる条件やクリティカル・パスを求めることができるため，本手法は誤動作の原因追跡や設計の改善にも有効である．また，本論文のアルゴリズムで直接シミュレーションできる回路は組合せ回路に限定されるが，文献11)と同様の手法により，非同期式順序回路の設計検証にも適用することができる．

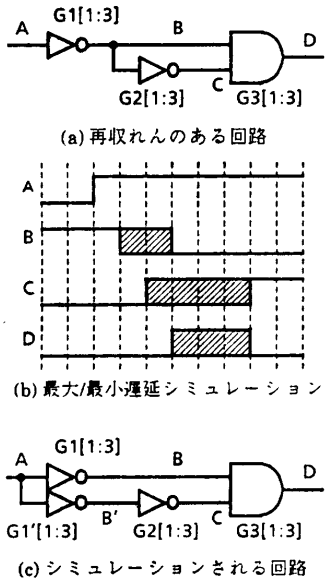
以下本論文では，2章でばらつきのある遅延のモデル化について考察し，3章で時間記号シミュレーションのアルゴリズムを示す．4章では時間記号シミュレーション・プログラムの実現とその設計検証への応用について述べる．

2. 遅延のばらつきのモデル化

2.1 最大／最小遅延モデルの問題点

現実の論理回路中の素子の遅延は，製造条件や使用環境の条件により異なってくる．このため，タイミングの微妙な回路では，素子遅延のばらつきによって誤動作が生じないかどうかを検証することが必要になる．

論理シミュレーションにおいてこのような素子遅延のばらつきを扱う方法としては，最大／最小遅延モデルによるシミュレーション[1]が提案されている．これは，**図1**のように，入力の変化に対する遅延の最小値から最大値までの信号値を不定値で表現することによって，遅延のばらつきをモデル化するものである．シミュレーションは，回路規模と入力信号値系列の長さに比例する時間で行えるが，再収れんの存在する回路においては，結果が悲観的になる場合があることが知られている[1]．例えば，図1(a)の回路においてAに立上り信号を印加すると，Dには実際には発生することのないスパイクの可能性が出力される（図1(b)）．大規模な回路では，このようにして発生した不定値が次々に伝播して，信号値の大部分を不定値としてしまうため，真に設計誤りがあるのかどうかを判定することが極めて難しくなってしまう．



(a) 再収れんのある回路

(b) 最大/最小遅延シミュレーション

(c) シミュレーションされる回路

図1 最大／最小遅延シミュレーションの悲観性
Fig. 1 Over pessimism in min/max delay simulation.

シミュレーション手法の改良により，単一の再収れん程度であれば，この悲観性を除去するものも存在する．しかし，遅延のばらつきを仮定した場合の組合せ回路の単一入力変化に対するハザード検出問題がNP困難であることが示されており[4]，最大／最小遅延シミュレーションの単純な改良では，この悲観性を解決できないと考えられる．

2.2 時間変数を用いたモデル化

図1のシミュレーションにおける悲観性は，G1で発生し分岐した遅延のばらつきの影響がG3で再収れんする際に，これが同じ源によるものであることを識別できず，結果的に図1(c)の回路をシミュレーションしていることに起因する．本論文では，遅延の値を変数で表現し，同じ遅延のばらつきによる影響を識別することにより，この問題を解決するというアプローチをとる．以下，この変数を時間変数と呼ぶ．遅延の最大値，最小値などの制限がある場合には，これを $\{3 \leqq d,\ d \leqq 5\}$ のように不等式集合で表す．この不等式集合を変数制限条件と呼ぶ．本論文では，この不等式として一般に一次不等式を許す．したがって，$\{d_1 \leqq 2d_2\}$ のように遅延時間の関係を記述することも可能である．この時間変数を用いて，様々な遅延のばらつきをモデル化することが考えられる．本論文では，次の3種類のモデル化[9]を考える．

1) 静的標準遅延 (static nominal delay)

1つの素子遅延に1つの時間変数を割り当てる．すなわち，遅延の値は未知ではあるが一定であると考えるモデル．

2) 静的立上り/立下り遅延 (static rise/fall delay)

立上り，立下りの遅延に対して異なる時間変数を割り当てるモデル．

3) 動的遅延モデル (dynamic delay)

信号値の変化ごとに時間変数を割り当て，遅延の値

が信号値の変化ごとに異なり得ることをモデル化したもの.

3. 時間記号シミュレーションのアルゴリズム

3.1 時間優先評価アルゴリズム

遅延を考慮した論理シミュレーションの基本的アルゴリズムとしては，イベント法[1]が広く用いられている．イベント法は回路中の信号線の信号値変化をイベントとしてとらえ，その伝播を計算することによりシミュレーションを行うものである．一般には，時刻を逐次進めながら，各時刻でのイベントの影響を計算していく，空間優先評価のアルゴリズム（文献3)の分類による．以下Sアルゴリズムと略す.）が用いられる.

遅延を時間変数で表した場合，シミュレーション時刻は一般に時間変数の一次式になる．この場合，Sアルゴリズムに基づくシミュレーションは可能ではあるが[5]，この方法では極めて多くの場合分けを行う必要が生じ，小規模な回路に対しても実現的な時間でシミュレーションを行うことは難しいと考えられる[5]．そこで，本論文では，時間優先評価アルゴリズム（Tアルゴリズム)[3]に基づく組合せ回路向きの計算法を提案する.

Tアルゴリズムは組合せ回路のシミュレーションに対しては単純で，効率的な計算法である[3),13)]．シミュレーションは，ゲートの出力信号値系列全体を（時間方向に）計算する，という操作をレベル順[1]にすべてのゲートについて行うことにより進められる.

Tアルゴリズムに基づくイベント法では，回路中の信号線の信号値系列を**図2**（a）のようなイベント（発生時刻と新信号値の組）の線型リストで表現する．ただし，発生時刻が$-\infty$である先頭のイベントを初期値イベント，最後尾のイベントを終了イベントと呼ぶ．終了イベントはこれ以降にイベントが発生しないことを表す（発生時刻は∞と考える)．ゲートの出力信号値系列の計算は，次のE1)～E2)を繰り返すことにより行われる.

E1) 入力イベント・リスト中の未評価のイベントのうち，時刻が最小のものを選ぶ.

E2) このイベントを評価し，出力を計算する．もし出力が変化すれば，新しいイベントを出力イベント・リストの最後尾に付加する.

図2（b）は，時間変数を含まない通常のシミュレーションにおける，ゲートの出力信号値系列の計算の例である．信号値系列AとBを入力とする遅延5の*and*ゲートの出力信号値系列Yは，X1)～X4)の手順で計算される.

X1) ①②より⑥を計算する.

X2) ③を評価する．出力は変化しないので，イベントの追加は行わない.

X3) ④を評価し，⑦を出力のリストに追加する.

X4) ⑤を評価し，⑧を出力のリストに追加する.

3.2 信号値系列の表現

シミュレーション時刻が時間変数を含む場合には，シミュレーション結果は時間変数の値の組合せによって異なり得る．本論文のアルゴリズムでは，これに対応するために，信号線の信号値系列を表現するデータ構造を線型リストからイベント木と呼ばれる有向木に拡張する.

イベント木は**図3**（a）のように，イベント節点，条件節点，およびこれらを結ぶ有向枝からなる．有向枝は

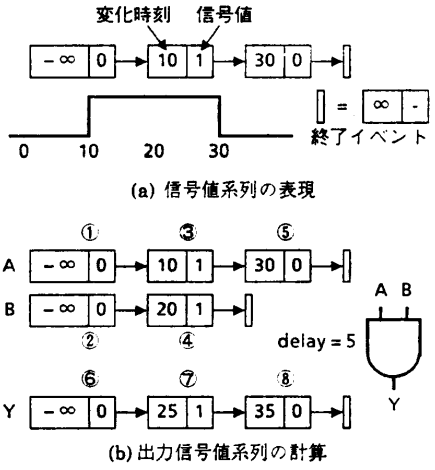


図2 時刻が定数の場合のTアルゴリズム
Fig. 2 T-Algorithm with time represented by constants.

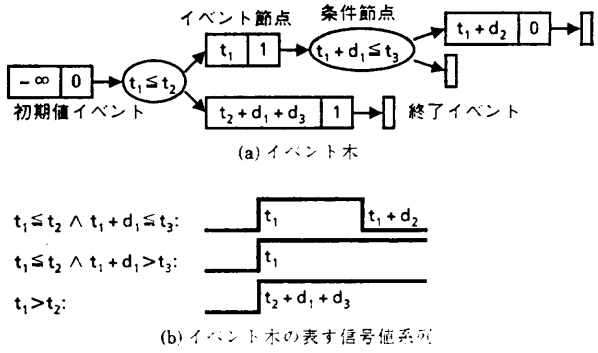


図3 イベント木
Fig. 3 Event tree.

イベントの発生順序を表す．初期値イベントは唯一であり，イベント木の根となる．終了イベントを除くイベント節点からは唯一の有向枝が出る．条件節点は時間変数の一次不等式γでラベル付けされている．条件節点からは，2つの有向枝が出ており，1番目（図では上側）の枝は時間変数がγを満たす場合，2番目（下側）の枝は時間変数が$\neg\gamma$を満たす場合に起こり得るイベントの系列を指す．γ，$\neg\gamma$をそれぞれ1番目の枝，2番目の枝の分岐条件と呼ぶ．また，根からある節点に至るパス上の分岐条件の積を，その節点のパス条件と呼ぶ．図3(a)のイベント木は，(b)に示す3通りのイベント系列を表している．

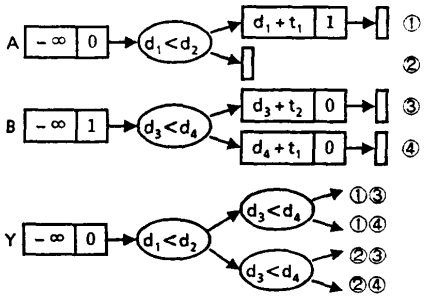


図 4 出力信号値系列の計算における場合分け
Fig. 4 Branching in computation of an output event sequence.

3.3 出力信号値系列の計算

出力信号値系列の計算は，基本的には時刻が時間変数を含まない場合と同様であるが，以下B1)，B2)に述べるような相違がある．

B1) ある入力信号線に次に起こり得るイベントは複数存在し得るので，それぞれが起こった場合について計算を行わなければならない．例えば，図4のように入力A，Bの次のイベントとしてそれぞれ①と②，③と④が起こり得るときには，①③，①④，②③，②④が起こるという組合せについて計算を行わなければならない．

B2) 各入力の次のイベントが一意であっても，時刻最小のものを一意に決めることができないことがある．図4において①④の組合せが選ばれた場合，d_1とd_4の大小が決定できなければ，それぞれのほうが小さい場合について計算を行わなければならない．

ただし，いずれの場合にもパス条件や変数制限条件から場合分けが不要になることがある．例えば，B2)の例において，変数制限条件に$\{d_1 \leqq 3, 5 \leqq d_4\}$が含まれていれば，必ず$d_1+t_1$のイベントが$d_4+t_1$より先に起こることがわかるので，場合分けは不要である．

```
function gateEval(節点 a, 節点 b) return(節点)
begin op(a, b, 変数制限条件, 未定義値, 未定義値, 未定義値) end gateEval;

function op(節点 a, 節点 b, 不等式集合 pc, 信号値 vy, 信号値 va, 信号値 vb)
   return(節点) .
begin
   if a が条件節点 then
   begin
      pc+:=pc∪{a.cond};
      pc-:=pc∪{¬a.cond};
      if       sat(pc+)∧¬sat(pc-) then return op(a.upper, b, pc, vy, va, vb)
      else if  ¬sat(pc+)∧sat(pc-) then return op(a.lower, b, pc, vy, va, vb)
      else if  sat(pc+)∧sat(pc-) then
         return cnode( a.cond,  op(a.upper, b, pc+, vy, va, vb),
                                op(a.lower, b, pc-, vy, va, vb))
   end
   else if b が条件節点 then a が条件節点の時と同様
   else if a, b ともにイベント節点 then
   begin
      pc+:=pc∪{a.time≦b.time};
      pc-:=pc∪{a.time>b.time};
      if       sat(pc+)∧¬sat(pc-) then return evalA(a, b, pc, vy, va, vb)
      else if  ¬sat(pc+)∧sat(pc-) then return evalB(a, b, pc, vy, va, vb)
      else if  sat(pc+)∧sat(pc-) then
         return cnode( a.time≦b.time,   evalA(a, b, pc+, vy, va, vb),
                                        evalB(a, b, pc-, vy, va, vb))
   end
end op;

function evalA(節点 a, 節点 b, 不等式集合 pc, 信号値 vy, 信号値 va, 信号値 vb)
   return(節点)
begin
   vy':=FUNC(a.val, vb);
   next:=op(a.next, b, pc, vy', a.val, vb);
   if(vy'≠vy)  then return enode(a.time+DELAY, vy', next)
               else return next
end evalA;

function evalB は evalA と同様;
```

図 5 ゲートの出力信号値系列計算のアルゴリズム
Fig. 5 Algorithm for computing an output signal sequence of a gate.

図5は静的標準遅延モデルの2入力ゲートの出力信号値系列の計算のアルゴリズムを記述したものである．記述の説明は以下のとおりである．

- 関数 gateEval は2つの入力に与えられたイベント木の根となる節点 a，b から出力のイベント木を計算し，その根の節点を返すものである．
- 関数 op は入力に与えられた2つの部分イベント木の根 a, b，変数制限条件とパス条件 pc，出力と入力の現時点での信号値 vy, va, vb から，a, b に対応する

出力の部分イベント木の根を返すものである．

● evalA, evalB は，根 a, b がともにイベント節点である2つの部分イベント木から，それぞれ a, b が先に発生した場合に得られる出力の部分イベント木の根を返す関数である．

● イベント節点 e の発生時刻，信号値，次の節点をそれぞれ，e.time, e.val, e.next で表す．また，e.time $=t$, e.val $=v$, e.next $=n$ であるようなイベント節点を enode(t, v, n) で表す．

● 条件節点 c の一次不等式を c.cond, c の1番目（上側）の節点，2番目（下側）の節点の有向枝に指される節点をそれぞれ，c.upper, c.lower と表す．また，c.cond $=i$, c.upper $=u$, c.lower $=l$ であるような条件節点を cnode(i, u, l) で表す．

● FUNC, DELAY はそれぞれ当該ゲートの論理関数，遅延に対応する時間変数を表す．

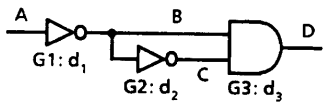
● sat(is) は不等式集合 is 中の不等式をすべて成立させる時間変数への割当てが存在するかどうかを判定する関数である．計算法については3.5節で述べる．

静的立上り／立下り遅延モデルの場合は，新しい出力値の計算結果によって立上り遅延，立下り遅延の時間変数を選択することが必要である．さらに，立上り遅延と立下り遅延を区別する場合には，時間変数を含まないシミュレーションの場合と同様，イベント取り消しの操作が必要になることがある[9]．動的遅延の場合には，出力が変化するごとに新しい時間変数を導入すればよい．ただし，この場合もイベント取り消しの操作が必要になる[9]．

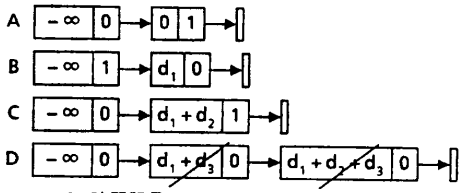
3.4 シミュレーション例

図6は図1と同様の回路のシミュレーションを時間記号シミュレーションにより行った例である．G3の計算において，CよりもBのイベントが先に起こることがわかるので，場合分けは生じない．その結果，Dにスパイクは発生しないという結論が得られる．

図7(b)は(a)の回路の入力Aに対して立上り信号を印加したときのシミュレーション結果である．これより，出力Eにスパイクの可能性があることがわかる．さらに，イベント木をたどれば，スパイクが発生する条件は $d_1<d_2 \wedge d_1+d_3<d_2$（すなわち $d_1+d_3<d_2$）であり，例えば d_1 を大きくすればスパイクを防止できるなどの情報を読み取ることもできる．

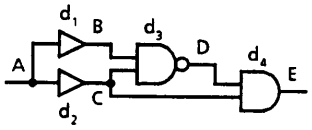


(a) 図2の回路

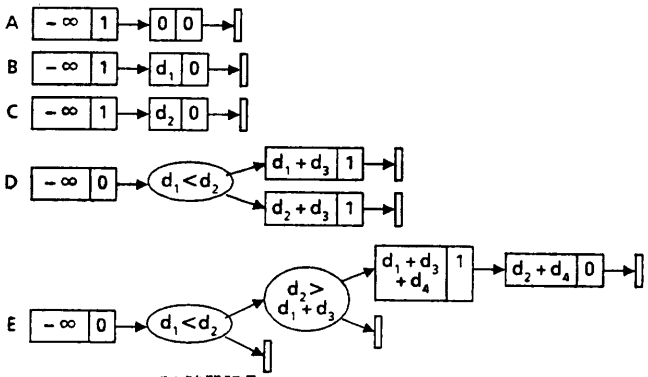


(b) 時間記号シミュレーションの結果

図6 時間記号シミュレーションの例（1）
Fig. 6 Example of time-symbolic simulation (1).



(a) 回路



(b) 時間記号シミュレーションの結果

図7 時間記号シミュレーションの例（2）
Fig. 7 Example of time-symbolic simulation (2).

3.5 記号式の扱い

3.3節で述べた処理を実行するためには，時刻や遅延の加算，不等式の簡単化，不等式集合の充足可能性判定などの記号処理が必要である．とくに，不等式集合の充足可能性判定は，パス条件と変数制限条件から起こり得ない場合分けを除外するために不可欠である．本論文では，シミュレーションの中で現れる不等式が一次不等式に限られることに注目し，解の存在判定を線型計画法に帰着して解く．すなわち，不等式集合を制約条件とし，これが可能基底を持つかどうかを判定することによりこの問題を解く．なお，この充足可能性の判定は，不等式を1つずつ追加するごとに行われる．そこで，毎回不等式集合に対して基底変換を行うのではなく，前回までの不等式集合をタブローの

形で保持しておき，追加される条件のみに基底変換を行うという方法により計算量の削減を図っている[7]．

4. 実現と応用

これまでに述べた方法に基づくシミュレータを Sun 3/60 の Unix (Sun OS 4.0) 上にC言語で実現した．線型計画法には simplex 法を用いた．本章では，時間記号シミュレーションの応用について述べる．

4.1 ハザードの検出への応用

3.4 節でも示したとおり，時間記号シミュレーションを用いれば，与えられた組合せ回路と入力信号値系列に対して，出力にスパイクが発生するかどうかを正確に判定できる．また，(スパイクのない) 期待値とシミュレーション結果を比較することにより，期待値どおりに回路が動作する条件を求めることができる．この比較に関しても，シミュレーションの結果解析系としてプログラムを実現しており，シミュレーション結果が期待値と一致する条件を一次不等式の積和形で求めることができる[8]．

4.2 非同期順序回路の検証

本論文のアルゴリズムで直接扱える回路は組合せ回路に限られるが，文献 11) と同様の方法によって，組合せ回路部分の性質を解析することにより，非同期順序回路の検証を行うことができる．

この検証の入力としては，

I1) 基本モード非同期式順序回路Mの状態遷移表，

I2) M のゲート・レベルの実現 C，

I3) 状態割当て，および状態変数と信号線の対応，

が与えられているものとする．検証の手順は次のとおりである（図 8 参照）．

V1) 状態変数に対応した信号線を切断し，組合せ回路 C′ を得る．

V2) すべての状態遷移に対し，V3)～V5) により状態遷移が正しく行われる条件を求める．

V3) 外部入力，外部出力，状態変数に現れると期待される信号値系列 X, Z_{exp}, Y_{exp} を状態遷移表より求める．Y_{exp} に関しては，状態遷移表より遷移前，遷移後の信号値がわかるので，これより適当に推測する．イベントの発生時刻が不明であれば時間変数で表す．

V4) X と Y_{exp} をそれぞれ外部入力と切断点に印加して時間記号シミュレーションを行い，外部出力，切断点の信号値系列 Z_{sim}, Y_{sim} を得る．

V5) Z_{sim} と Z_{exp}, Y_{sim} と Y_{exp} を比較し，これらが一致する条件を求める．これが正常に遷移が起こる条件である．

ただし，V3) において Y_{exp} を推測しているが，これ以外の信号値系列でも正常な遷移が起こる可能性はある．したがって，上記の方法で得られる条件は，正常動作のための十分条件である．

図 9 にTフリップ・フロップの検証例を示す．(a)，(b)はそれぞれ状態遷移表，回路の実現例である．状態割当ておよび状態変数と信号線の対応は，状態遷移表に与えられている．(c)は入力X，出力 Z (Y2)，および状態変数 Y1 に期待される信号値系列をタイムチャートで示したものである．Y1 の期待値としては，スパイクなしに状態遷移が起こるものを求めている．また，Xの変化時点から Z, Y1 の変化時点までの遅

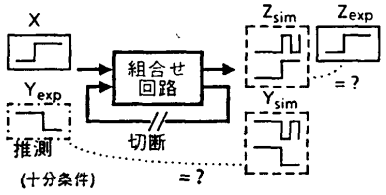
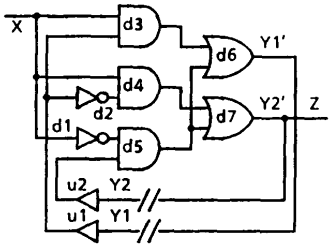


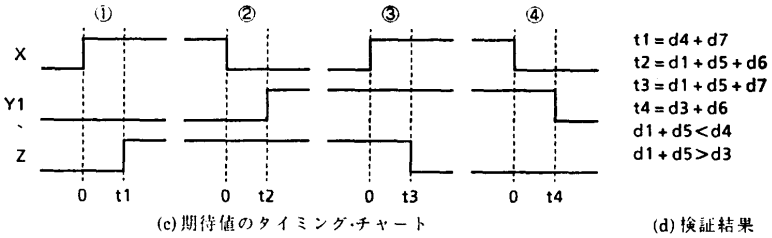
図 8 非同期式順序回路の検証法
Fig. 8 Method of the verification of asynchronous sequential circuits.

Y1 Y2	Y1′ Y2′ X=0	X=1	Z
0 0	(0 0)	0 1	0
0 1	1 1	(0 1)	1
1 1	(1 1)	1 0	1
1 0	0 0	(1 0)	0

(a) 状態遷移表



(b) ゲート・レベルの設計



(c) 期待値のタイミング・チャート　(d) 検証結果

図 9 Tフリップ・フロップの検証
Fig. 9 Verification of a T-flipflop.

延が不明なので，これを $t1$〜$t4$ の時間変数で表している．

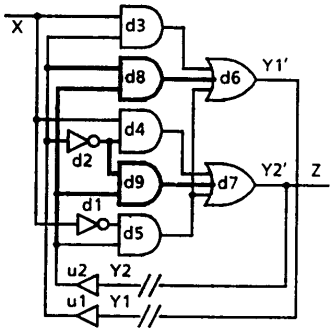
シミュレーションを行い，期待値との比較を行うと，(d)のような条件集合が出力される．上方の等式は，$t1$〜$t4$ の値を表しており，これより各遷移のクリティカル・パスを知ることができる．下方の不等式は，回路が期待どおりにTフリップ・フロップとして動作する条件を表している．条件中にはフィードバック線の遅延を表す時間変数 $u1$, $u2$ が含まれていない．これより，この設計は組合せ回路中の遅延のばらつきによって誤動作を起こす可能性があり，しかも，その誤動作をフィードバック線の遅延を調整することによって回避することはできない，ということがわかる．

図 10 は図9の組合せ回路にハザード防止用のゲートを追加して，同じ検証を行った結果である．この設計では，正常動作の条件はいずれも $u2$ を含んでおり，$u2$ を大きくすることによって誤動作を防げることがわかる．また，実現は行っていないが，線型計画法を用いれば，回路が正常に動作する $u2$ の最小値を求めることも可能である．さらに，等式より，ハザード防止用のゲートを追加したことによるクリティカル・パスの変化はないことも読み取れる．

4.3 処理性能の評価

遅延のばらつきまでを考慮した組合せ回路のハザード検出問題がNP困難であることよりやむを得ないことであるが，時間記号シミュレーションの最悪の計算量は，少なくとも回路規模および入力信号値系列の長さの指数に比例すると考えられる．時間記号シミュレーションは，時間を代数式で表現し，遅延の可能な大小関係の数え上げを行うものである．ハザードの有無を問題にする回路では，この大小関係の総数がそれほど大きくならないことが多く，前節のような小規模な回路であれば，十分高速にシミュレーションできる．しかし，大部分が同期式順序回路として設計された大規模な回路に，本手法をそのまま適用することは難しい．同期式順序回路部分は別な方法で検証し，タイミングが問題になる部分回路に限って本手法を適用するという使い分けが必要であると考える．

使用計憶量に関しては，本論文のアルゴリズムでは，イベント木の大きさが，最悪の場合，回路規模と入力信号値系列の指数に比例して大きくなり，これが支配的となる．不等式集合の充足可能性判定を解くためのタブローの大きさは，時間変数の数と不等式集合の大きさの積に比例するが，この大きさも無視できない．現在，8 MB の実記憶で現実的な時間でシミュレーションが行えるのは100ゲート程度までである．



(a) ゲート・レベルの設計

t1 = d4 + d7
t2 = d1 + d5 + d6
t3 = d1 + d5 + d7
t4 = d3 + d6
d3<u2 + d8 + d1 + d5 + d7
d1<u2 + d4 + d7

(b) 検証結果

図 10 ハザード防止回路付きTフリップ・フロップの検証
Fig. 10 Verification of a T-flipflop with a hazard detection circuit.

5. おわりに

新しいタイミング検証のアプローチとして時間記号シミュレーションを提案し，そのアルゴリズム，応用の可能性について述べた．時間記号シミュレーションは，悲観性のない正確なシミュレーションを可能にするばかりではなく，回路が正常に動作する条件やクリティカル・パスの情報などを求めることができるため，設計検証ばかりでなく，設計の改善にも応用が期待できる．

今後の課題としては，組合せ回路以外の回路への適用の拡大，計算時間の削減があげられる．フィードバック線を含む回路を扱うためにはSアルゴリズムに基づく計算法を開発する必要があるが，この際に，文献12) の手法のように無駄な場合分けを避ける方法を考えることが不可欠である．計算時間に関しては，問題の計算量そのものが大きいため，現実的な時間で大規模な回路のシミュレーションを行うためには，従来の最大／最小遅延シミュレーションと組み合わせる，あるいは不定値を利用して場合分けの数を減らすなどの，近似計算法が必要であると考える．

謝辞 本研究の遂行にあたり，ご討論・ご助言いただいた本学電子工学教室の安浦寛人助教授，情報工学教室の平石裕実助教授，高木直史博士，荻野博幸氏，

金原弘明氏はじめ，矢島研究室の諸氏に感謝します．
なお，本研究は一部文部省科学研究費補助金による．

参考文献

1) Breuer, M. A. and Friedman, A. D.: *Diagnosis & Reliable Design of Digital Systems*, p. 308, Computer Science Press, California (1976).
2) Carter, W. C., Joyner, W. H. and Brand, D.: Symbolic Simulation for Correct Machine Design, *Proc. ACM/IEEE 16th Design Automation Conf.*, pp. 280-286 (Jun. 1979).
3) 石浦菜岐佐，安浦寛人，矢島脩三：時間優先評価アルゴリズムによる論理シミュレーションの高速化，情報処理学会論文誌，Vol. 26, No. 3, pp. 459-466 (1985).
4) 石浦菜岐佐，安浦寛人：時間モデルとハザード検出問題の計算量の関係について，電子情報通信学会技術研究報告，COMP 88-21, pp. 29-37 (Jun. 1988).
5) 石浦菜岐佐，矢島脩三：時間記号シミュレーションについて，情報処理学会研究報告，87-DA-40-16, pp. 103-110 (Dec. 1987).
6) Ishiura, N., Takahashi, M. and Yajima, S.: Time-Symbolic Simulation for Accurate Timing Verification of Asynchronous Behavior of Logic Circuits, *Proc. ACM/IEEE 26th Design Automation Conf.*, pp. 497-502 (Jun. 1989).
7) 高橋瑞樹，石浦菜岐佐，矢島脩三：時間記号論理シミュレータの高速化と性能評価，第 37 回情報処理学会全国大会論文集，3 U-3, pp. 1759-1760 (Sep. 1988).
8) 高橋瑞樹，石浦菜岐佐，矢島脩三：時間記号論理シミュレーションの結果解析系，情報処理学会研究報告，88-DA-44-2, pp. 9-16 (Oct. 1988).
9) 高橋瑞樹，石浦菜岐佐，矢島脩三：時間記号論理シミュレーションにおける遅延モデルの拡張，第 38 回情報処理学会全国大会論文集，5 S-7, pp. 1367-1368 (Mar. 1989).
10) 藤本徹哉，野田浩明，神戸尚志：制御信号系にもとづくタイミング検証，情報処理学会研究報告，87-DA-40-11, pp. 73-78 (Dec. 1987).
11) 木村晋二，羽根田博正，矢島脩三：正則表現論理シミュレーション手法に基づく非同期順序回路の検証，電子情報通信学会論文誌 (D)，Vol. J 71-D, No. 9, pp. 1786-1796 (Sep. 1988).
12) Yoneda, T., Nakade, K. and Tohma, Y.: A Fast Timing Verification Method Based on the Independence of Units, *Proc. IEEE 19th Int. Symposium on Fault-Tolerant Computing*, pp. 134-141 (Jun. 1989).
13) Ishiura, N., Yasuura, H. and Yajima, S.: High-Speed Logic Simulation on Vector Processors, *IEEE Trans. Computer-Aided Design*, Vol. CAD-6, No. 3, pp. 305-321 (May 1987).

(平成元年 8 月 31 日受付)
(平成 2 年 9 月 11 日採録)

石浦菜岐佐（正会員）

昭和 36 年生．昭和 59 年京都大学工学部情報工学科卒業．昭和 61 年同大学院修士課程修了．昭和 62 年 1 月より京都大学工学部助手．論理回路の CAD/DA の研究に従事．電子情報通信学会会員．

高橋　瑞樹（正会員）

昭和 38 年生．昭和 61 年京都大学工学部情報工学科卒業．平成元年同大学院修士課程修了．同年シャープ(株)入社．現在，同社生産技術開発センターにて論理回路の高位合成の研究に従事．在学中は，論理回路のタイミング検証の研究に従事．

矢島　脩三（正会員）

昭和 8 年生．昭和 31 年京都大学工学部電気工学科卒業．同大学院博士課程修了．工学博士．昭和 36 年より京都大学工学部に勤務，昭和 46 年情報工学科教授．昭和 35 年京都大学第一号計算機 KDC-1 を設計稼動，以来，計算機，論理設計，オートマトン等の研究教育に従事．著書は「電子計算機の機能と構造」(岩波，57 年) 等．本学会元常務理事，元会誌編集委員 (地方)，元 JIP 編集委員．電子情報通信学会元評議員およびオートマトンと言語研専元委員長，North-Holland 出版 元 IPL 編集委員，IEEE Senior Member.