数理解析研究所講究録
第556巻 1985年 104-115

論理回路機能の時間的関係の記述と検証

京都大学工学部　木村 晋二　(Shinji Kimura)
京都大学工学部　矢島 脩三　(Shuzo Yajima)

1. まえがき

論理回路の大規模化に伴い、誤りのない設計を支援するための方法として、論理回路の形式的な検証法に対する要求が大きくなってきた。論理回路の設計検証が古典的なプログラムの検証と異なるのは、論理回路の機能記述において基本的である時間の経過を系列により扱うことである[1]-[6]。

系列を扱う手段としての有限オートマトンは、同期式順序回路の良いモデルであるが、非同期式順序回路や遅延時間を持つゲートの組合せにより構成された論理回路に対しては、記述が複雑になったり、状態数の爆発的増加を引き起こしたりするので、余り良いモデルとは言えない。また、タイミングチャートなどのように相対時間を用いた動作の記述も、有限オートマトンでは扱いにくい。この原因は、論理回路の遅延時間が本来遠隔的な関係を表すものであるのに対し、有限オートマトンはあくまで次状態のみを考えているからであると考えられる。そこで本報告では、遠隔的な時間の関係を記述するための手法について考察し、各時間で端子がとる値を長さ2の系列の集合により制限することに基づいた記述法を提案した。例えば、{00, 11}は変化しないことを表し、{01, 10}は変化することを表す。制約を与えたい時間は基準時間からの変位により表され、基準となる時間もこの長

さ2の系列集合により表される。また、検証への適用に関して、この記述法を用いて記述された系列集合間の包含判定問題が可解であることを示した。さらに記述法に制限を加えることにより、包含判定が記述量の多項式に比例した計算量で効率的に行えることを示した。

2章では、制約式と呼ぶ系列集合の記述法と、記述された系列集合の持つ性質について述べる。3章では、実際の記述例を示す。但し、以下で用いる用語の定義は文献[7]に従う。

2. 論理回路の機能の記述と検証

2.1. 論理回路の時間的関係の記述法

論理回路の機能は、離散時間及び離散値の仮定の下では、入力アルファベットをΣ_I、出力アルファベットをΣ_Oとして、Σ_I^*からΣ_O^*への関数fと考えることができる。但し、論理回路の出力は未来の入力(まだ入力されていないもの)により影響を受けてはならないので、fは以下の条件を満たす必要がある。

(条件)　$x\in\Sigma_I^*$, $y\in\Sigma_O^*$ に対し、$f(x)=y$である時には、$|x|=|y|$かつ、1以上$|x|$以下の任意のiに対し、$f(x[1,i])=y[1,i]$である。(x[1, i]は系列xの1番目からi番目までの記号からなる部分系列を意味する。)

関数を直接扱うよりも系列集合として扱う方が便利であるので、論理回路の機能を表す関数fを、$(\Sigma_I\times\Sigma_O)^*$の部分集合

$$\{(x[1], y[1])\cdot(x[2], y[2])\cdot\ldots\cdot(x[n], y[n]) \mid f(x)=y, |x|=n, x\in\Sigma_I^*, y\in\Sigma_O^*\}$$

により表す。$|x|$は系列xの長さを表す。逆に系列集合を一つ与えれば、論理回路の機能を表す関数を与えたことになる。これを論理回路の機能を表す系列集合と呼ぶ。先程述べた条件は、論理回路の機能を表す系列集合をSとすると、「$x\in S$なら

ば、1以上|x|以下の任意のiに対し、x[1, i]∈S」と表せる。一般的には、関数fの与え方により、ここで定義された系列集合は正則集合とならない可能性があるが、現実には正則もしくはその部分クラスに限定しても良いと思われる。

[補題1]　Sが正則集合である時、Sに含まれる各系列xの先頭から始まる全ての部分系列からなる集合$\{x[1, i] \mid x \in S, 1 \leq i \leq |x|\}$は正則集合である。

[証明]　補題の仮定より、Sを受理する決定性有限オートマトン$M=(Q, \Sigma, \delta, q_0, F)$が存在する。但し、δは通常の意味で$Q \times \Sigma^*$からQへの関数に拡張されているとする。この時、Qの部分集合F'を$\{q \mid \exists x \in \Sigma^*.\delta(x, q) \in F\}$により定義すると、有限オートマトン$M'=(Q, \Sigma, \delta, q_0, F')$は系列集合$\{x[1, i] \mid x \in S, 1 \leq i \leq |x|\}$を受理する。なぜならば、Mに受理される任意の系列xに対し、x[1, i]により到達する状態$\delta(x[1, i], q_0)$はF'に含まれ、よってM'により受理されるからである。　Q.E.D.

論理回路の機能記述は、3つ組(I,O,F)により定義される。ここで、Iは入力端子の有限集合、Oは出力端子の有限集合である。端子は、端子名AとアルファベットΣの2つ組(A, Σ)により定義される。Σは、この端子のとる値の集合を意味する。また混乱が生じないかぎり、端子と端子名を同一視する。Fは、I∪Oに関する制約式の集合である。端子の有限集合Tに関する制約式は、以下のように定義される。

[制約式の定義]

(1)　T中の端子$A=(A, \Sigma_A)$と$\Sigma_A{}^2$の部分集合S_Aの2つ組(A, S_A)は制約式である。以下ではこの形の制約式を値式と呼ぶ。例えば$(A, \{01, 10\}), (B, \{00, 11\})$は値式である。

(2)　f_1, f_2が制約式であり、非負整数mと非負整数または∞(任意の整数kに対しk≤∞)のnに対し、m≤nならば、$(f_1 \rightarrow [m, n]\ f_2)$及び$(f_1 \rightarrow <m, n> f_2)$は制約式である。□

Tに関する制約式を満たす系列はモデル(Σ, Θ)上で定義される。ここでΣはアルファベットを表し、ΘはΣからアルファベットΣ_A(Σ_AはTに含まれる端子のアル

ファベットとする)への関数θ_Aの集合を表す。以下では、系列$x(\in\Sigma^*)$が制約式fを満たすことを$x\models f$と表す。

(1) 値式(A, S_A)を満たすΣ上の系列は、Σ_Aを端子Aのアルファベットとして、

$\{x\in\Sigma^* \mid |x|=1$かつ$\theta_A(x)\cdot\Sigma_A\cap S_A\neq\emptyset$であるか、$|x|\geq 2$かつ$\theta_A(x[1])\cdot\theta_A(x[2])\in S_A\}$

により定義される。

(2) 制約式$f_1\rightarrow[m, n]f_2$ $(f_1\rightarrow<m, n>f_2)$を満たす系列集合は、

$\{x \mid x\in\Sigma^*$かつ、$x[i, |x|]\models f_1$であるような全てのiに対し、m以上n以下の全てのjで

$(<m, n>$:あるjで)、$x[i+j, |x|]\models f_2$または$i+j>|x|$である$\}$

により定義される。これは、f_1が成立する時間を基準として、m以上n以下の任意の時間で$(<m, n>$:ある時間で)f_2が満たされているような系列を表す。また、この制約式を満たす系列の内で、$x[i, |x|]\models f_1$であるようなあるiに対し、m以上n以下のあるjが存在し$x[i+j, |x|]\models f_2$であるようなものを特に強い意味で制約式を満たす系列と呼ぶ$(<m, n>$の場合も同様)。□

制約式が成立するかどうかは、先頭の値式が成立する時間から考えなければならないので、先頭の値式が成立する時間をこの制約式の基準時間と呼ぶ。

端子集合$\{(A, \{0, 1\}), (B, \{0, 1\}), (C, \{0, 1\})\}$に関する制約式の例と、それらの制約式を満たす系列集合のタイミングチャートによる記述を図1に示す。但し、唯一の要素からなる集合{a}は、単にaと表す。

次に、制約式の集合Fを満たす系列を定義する。これは、実際の論理回路の機能を表す場合には一つの制約式では記述能力が十分でないからである。制約式の集合F中の全ての制約式fに対し$x\models f$であるような系列xをFを満たす系列と呼び、$x\models F$と表す。制約式の集合Fに対し、系列集合$\{x \mid x\models F\}$で論理回路の機能を表す。

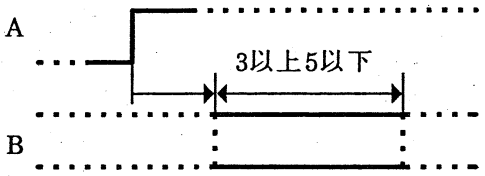

図1(a). (A,01)→<3,5>(B,{00,11})

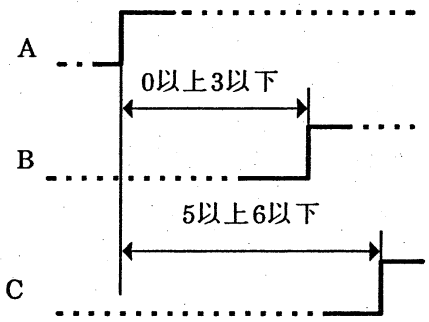

図1(b). ((A,01)→<0,3>(B,01))→<5, 6>(C,01)

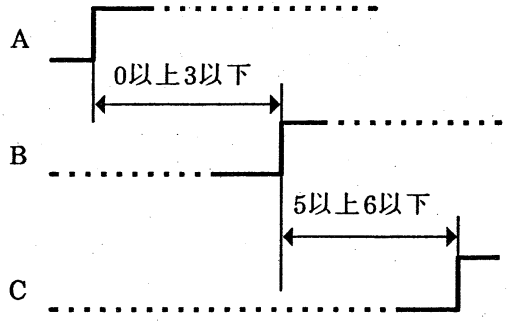

図1(c). (A,01)→<0, 3>((B,01)→<5, 6>(C,01))

図1. 制約式を満たす系列集合のタイミングチャートによる記述

2.2. 論理回路機能の検証について

論理回路の設計においては、まずその仕様を制約式集合により記述する。仕様に誤りがないことの検証として、各制約式が2.1で述べた条件を満たすかどうかの判定と、制約式集合を満たす系列が存在するかどうかの判定(以下では無矛盾性判定と呼ぶ)とが重要である。また、この仕様に対する論理回路の実現が与えられた場合には、実現により生じる制約式集合が仕様である制約式集合に含まれることの判定(包含判定)を行うことにより、実現が仕様を満たすことの検証を行うことができる。これらは、設計検証の3つの段階を表している。

制約式fが2.1で述べた条件を満たすかどうかについて考察する。例えば、

((A, 01)→<3, 4>(B, 01))→<1, 2>(C, 01)および

((A, 01)→<5, 7>(B, 10))→[4, 6]((C, 01)→<1, 2>(D, 10))は、2.1の条件を満たさない。以下にこの条件の判定を行う再帰的なアルゴリズムを示す。

(1) fが値式ならば明らかに2.1の条件を満たす。この時min(f)=max(f)=0とする。

(2) fが($f_1 \rightarrow [m, n]\ f_2$)または($f_1 \rightarrow <m, n> f_2$)ならばf_1およびf_2が条件を満たすかどうかを判定する(再帰)。もし$\max(f_1) \leq m + \min(f_2)$ならばfは条件を満たす。この時、$\min(f) = \min(f_1) + m + \min(f_2)$, $\max(f) = \max(f_1) + n + \max(f_2)$とする。また、$\max(f_1) \leq m + \min(f_2)$が成立しないならばfは条件を満たさない。この時、$\min(f) = \max(f) = \perp$(未定義を表す)とする。また任意のiに対し$i + \perp = \perp + i = \perp$とし、$\perp \leq i$および$i \leq \perp$のいずれも成立しないとする。□

アルゴリズム中のminおよびmaxは、基準時間から制約式が成立するまでの最小および最大の時間を表している。再帰呼び出しの回数は、制約式に含まれる→の数の2倍に等しいので、制約式の長さで上から押えられる。

制約式集合が無矛盾であるとは、制約式集合中の全ての制約式を強い意味で満たす系列が存在することを言う。例えば、(a) {(A, 01)→[1, 3](A, 00)}、(b) {(A, 01)→[1, 5](B, 10)}、(c) {(A, 01)→[2](B, 01), (A, 01)→[1, 3](B, 00)}は、矛盾している。(a)では端子Aに01が生じる時、次の時間で0(時間1から3で00)であることを示しているが、01となる以上次の時間の値は1であり(値式は長さ2の系列の部分集合であるので、各時間で次の時間の値も指定していることに注意)矛盾している。(b)は(a)と同様である。(c)では時間2に端子Bに対して01と00の相異なる制約を表す二つの制約式から成り、これらを同時に満たす系列は存在し得ない。

制約式集合が矛盾するのは、(1) 同じ時間に同じ端子に対し二つ以上の値式$(A, S_{A_1}), (A, S_{A_2}), \ldots, (A, S_{A_n})$ $(n \geq 2)$が存在し$S_{A_1} \cap S_{A_2} \cap \ldots \cap S_{A_n} = \varnothing$であるか、(2) 時間i及びi+1で同じ端子に対する値式$(A, S_A), (A, S_A')$が存在しS_Aの各要素abに対しS_A'中にbで始まる要素がないか、のいずれかである。

制約式の集合FとGおよびモデル(Σ, Θ)が与えられた時に、各々の制約式集合を満たす系列集合間の包含判定問題$\{x \in \Sigma^* | x \models F\} \supseteq \{y \in \Sigma^* | y \models G\}$を$F \supseteq G$と表す。

制約式集合の無矛盾性判定および、制約式集合間の包含判定問題の可解性は以下の定理から保証される。

[定理1]　制約式集合を満たす系列集合を受理する有限オートマトンを構成することができる。

[略証]　制約式集合を満たす系列集合は制約式集合の中の各制約式を満たす系列集合の共通部分であり、これを受理する有限オートマトンは各制約式を満たす系列集合を受理する有限オートマトンの直積により構成することができる。よって、単一の制約式に対する構成法を示せば良い。

(1) 値式(A, S_A)を満たす系列集合は、$\{x\in\Sigma^* \mid |x|\geqq 2$かつ$\theta_A(x[1])\cdot\theta_A(x[2])\in S_A\}$なる正則集合である(補題1の性質に注意)。

(2) 制約式$f_1\rightarrow[m, n]\ f_2$または$f_1\rightarrow<m, n>f_2$を満たす系列集合を受理するかどうかの判定を行う有限オートマトンは、f_1, f_2が満たされることの判定を行う有限オートマトンを用いて、再帰的に構成される。

f_1の先頭の値式が成立するどうかを各時間で行う。成立した時間が基準時間であるから、そこからm時間からn時間だけ待ち(高々n状態用いれば良い)、f_2が満たされることの判定を行う有限オートマトン(M_2)を初期化して実行する。同時に、f_1の残りの部分が成立するかどうかを判定し、成立するならばf_2が強い意味で満たされるまでM_2を動作させる。またf_1の残りの部分が成立しないならば、M_2を初期化する。もしも$n\neq\infty$ならば、M_2は同じ時間には高々有限個(n-m個)しか存在しない。なぜならば、初期化した有限オートマトンは一定時間内に受理·非受理を判定して初期状態に戻り、次に先頭の値式が成立する時には同じ有限オートマトンを用いることができるからである。また$n=\infty$の場合でも、M_2は高々有限個で良い。なぜならば、最初にm以上になったものだけが意味を持つので、それ以外の有限オート

マトンは初期化してもよいからである。よって、このような動作を行う有限オートマトンを構成することができる。□

[系1]　制約式集合Fの無矛盾性判定問題は可解である。□

[系2]　制約式集合F,Gに関する包含判定問題F⊇Gは可解である。□

この証明に従い構成される有限オートマトンの状態数は非常に大きくなるので、この有限オートマトンを用いて包含判定等を行うことは実用的でない。以下では、特殊な場合に包含判定が効率的にできることを示す。

[定理2]　制約式中の値式の数を2に制限すると、制約式集合FとGに対する包含判定F⊇Gを、F,Gの記述量(|F|,|G|)の多項式に比例した時間で行うことができる。

[略証]　F⊇Gであるための必要十分条件は、Fに含まれる任意の制約式fに対し、{x|x⊨f}が{x|x⊨G}を含むことである。もしもF⊇Gであるならば{x|x⊨f}⊇{x|x⊨F}であり、明らかに{x|x⊨f}は{x|x⊨G}を含む。逆に、任意の制約式fに対して、{x|x⊨f}が{x|x⊨G}を含むならば、{x|x⊨f}のfに対する共通集合も{x|x⊨G}を含み、F⊇Gである。さらに、A→(m, n)B ((m, n)は[m, n]または<m, n>のいずれかを示す)の形の制約式に対し、Aが成立してからBが成立するまでに複数回Aが成立することは考える必要がない。なぜならば、値式の数が2個に限定されているので、何回かAが成立するとBは成立しなくなるというような複雑な制約は制約式集合を用いても記述できないからである。

F中の制約式f=A→(m,n)Bに関して、値式Aが成立してから値式Bが成立するまでは、値式Aが成立しないという仮定のもとで、制約式fを満たす系列集合を受理する決定性有限オートマトンMを構成する。Mは、A, Bを満たす長さ2の系列を受理する決定性有限オートマトンをM_A, M_Bとし(M_A, M_Bが構成できることは明らか)、M_AからM_Bの間にm状態挿入し、さらにM_Bをmからnの間はいつBを満たす系

列がきても良いように拡張し、Bを満たす系列を受理したら初期状態に戻るようにすることにより、構成することができる。Mの状態数は、M_Aの状態数をp、M_Bの状態数をqとして、m+p+n×qの整数倍で上から押えられる。

一方、{x|x⊨G}を受理する非決定性有限オートマトンであるが、各制約式を満たす系列集合を受理する有限オートマトンはfの場合と同様に構成すれば良い。それらの共通集合に関しては、各制約式の前の値式まとは後の値式が同じである場合には、非決定的にあらゆる場合への遷移を付加する。その結果構成された有限オートマトンに関し、状態間が連結でないものは異なる有限オートマトンとして扱う。これらのオートマトンは独立な制約に関係するものであり、各々の制約の前の値式は任意の時間で生じるので別々に扱っても良い。各有限オートマトンの状態数は、以前と同様記述量の多項式のオーダである。また、有限オートマトンの数も記述量のオーダを越えない(制約式の数以下である)。

後は、fを満たす系列集合がGを満たす系列集合を含むかどうかの判定を有限オートマトンを用いて行えば良い。この判定は、fに対する有限オートマトンM_fが決定性であることから、各々のオートマトンの状態数の積に比例した計算量で判定できることがわかる(M_fの最終状態でないものを最終状態とするオートマトンM_f'と、M_Gの直積オートマトンを構成し、それらの受理集合の共通集合が空であるかどうかの判定である)。よって、全体として記述量の多項式に比例した計算量で包含判定ができる。□

3. 記述例

本章では、例として遅延素子の記述を行う。

(記述例1)　入力端子が(A,{0,1})、出力端子が(Y,{0,1})である遅延素子が5単位時間の純粋遅延(pure delay)を持つとすると、この素子の機能は、

{(A, 01)→[5](Y, 01), (A, 10)→[5](Y, 10), (A, {11, 00})→[5](Y, {11, 00})}

なる制約式の集合により表される。この集合の第一の制約式(A, {01})→[5](Y, {01})は、端子Aにおいて01なる変化が生じた時5単位時間後にYに01なる変化が生じるべきことを示しており、第二の制約式はその逆の変化に関する表明である。また、第三の制約式は、入力が変化しない場合の表明である。

(記述例2)　入力端子が(A,{0,1})、出力端子が(Y,{0,1})である遅延素子が5単位時間の慣性遅延(inertial delay)を持つとすると、記述例1と同様

{((A, 01)→[1,4](A, 11))→[5,9](Y,{01,11}), ((A, 10)→[1,4](A, 00))→[5,9](Y,{10, 00}),

((A, {00, 11})→[1, 3](A, {00, 11}))

→<4, ∞>(((A, {01, 10})→<1, 4>(A, {01, 10}))→[1, 9](Y, {00, 11})),

((A, {00, 11})→[1, 3](A, {00, 11}))→[5, 9](Y, {00, 11})}

なる制約式集合により機能が記述される。尚、図2にタイミングチャートによる記述を示す。この集合中の最初の制約式((A,{01})→[1,4](A,{11}))→[5,9](Y,{01,11})は、端子Aにおいて01の変化が生じている時間を基準にして、その後5単位時間の間はAで変化が生じないなら、基準時間の後6単位時間から10単位時間までは端子Yで1が確定することを示している。また、第三の制約式は、端子Aが5単位時間の間変化しないならば、それ以後のAの初めての変化から5単位時間内に変化が起こる時、最初の変化から10単位時間の間はYの値が変化しないことを示しており、短い幅のパルスは出力側へ影響を及ぼさないという性質を表している。

5. あとがき

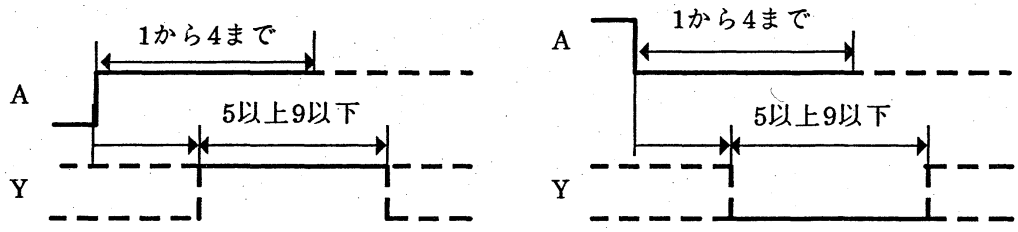

図2(a). ((A, {01})→[1, 4](A, {11}))→[5, 9](Y, {01, 11}),
((A, {10})→[1, 4](A, {00}))→[5, 9](Y, {10, 00})

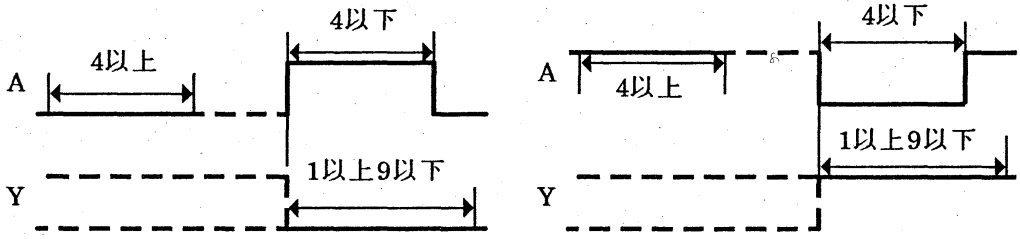

図2.(b) ((A, {00, 11})→[1, 3](A, {00, 11}))
→<4, ∞>(((A, {01, 10})→<1, 4>(A, {01, 10}))→[1, 9](Y, {00, 11}))

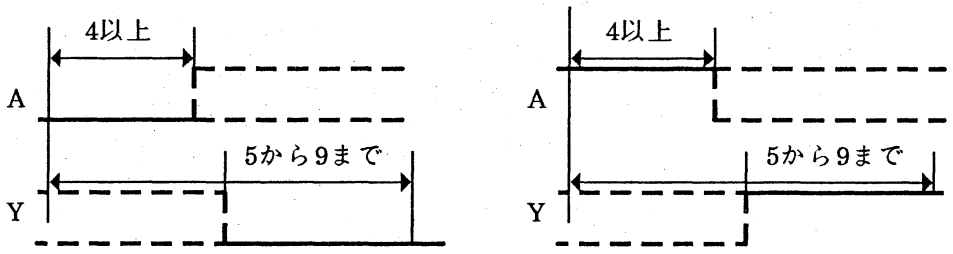

図2.(c) ((A, {00, 11})→[1, 3](A, {00, 11}))→[5, 9](Y, {00, 11})

図2.慣性遅延を持つ遅延素子

本報告では、論理回路の時間的関係を表す系列集合の記述法について述べた。ここで提案した記述法は、タイミングチャートなどを用いて非形式的に記述されることの多い、論理回路の非同期的な動作の記述に適している。また、この記述法で記述したものは系列集合により厳密な意味が与えられるので、論理回路の形式的検証に用いることができる。

ここで提案した記述法の性質として、これで表される系列集合を受理する有限オートマトンが構成できることを示した。このことから、これらの系列集合間の包含判定等は決定可能であることがわかる。さらに、記述法に制限を加えること

により包含判定における計算量が記述量の多項式のオーダーになることを示した。今後は、ここで提案した記述法で表された系列集合の包含判定における計算量のより厳密な評価を行いたいと考えている。

謝辞

日頃、御議論頂く本学平石裕実博士ならびに安浦寛人博士に感謝します。

参考文献

[1] Bochmann, G. V., "Hardware Specification with Temporal Logic: an Example," IEEE Transactions on Computers, vol. C-31, no. 3, pp. 223-231, 1982.

[2] 房岡、世木、高橋、「Temporal LogicによるVLSIの動作の記述と推論について」、信学技報、AL83-22,1983.

[3] 藤田、田中、元岡、「ハードウェア状態遷移表現のPrologによる検証」、情報処理学会論文誌、第25巻第4号、pp.647-654、昭和59年7月.

[4] Mishra, B., and Clarke, E. M., "Automatic and Hierarchical Verification of Asynchronous Circuits Using Temporal Logic," CMU Report, CMU-CS-83-155, 1983.

[5] Moszkowski, B., "A Temporal Logic for Multi-level Reasonig about Hardware," Technical Report of Stanford University, STAN-CS-82-952, 1982.

[6] 本橋、鈴木、他、「有限長及び無限長正規言語の記述と検証に関する一考察」、信学技報、AL83-3,1983.

[7] Hopcroft, J. E., and Ullman, J. D., "Introduction to Automata Theory, Languages, and Computation," Addison-Wesley, 1979.